UNIVERSIDAD CARLOS III

Escuela Politécnica Superior

Grado en Ingeniería Informática

DESARROLLO DE PORTAL WEB PARA ARTÍCULOS DE SEGUNDA MANO

TRABAJO FIN DE GRADO

Autor: Álvaro Cedillo Corral

Tutor: Jesús Hernando Corrochano

Febrero de 2014

Título: Desarrollo de portal Web para artículos de segunda mano

Autor: Álvaro Cedillo Corral

Tutor: Jesús Hernando Corrochano

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Trabajo Fin de Grado el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

AGRADECIMIENTOS

En primer lugar, quisiera dar las gracias a mis padres por el apoyo que me han dado, y por el esfuerzo realizado para que haya podido tener la educación que he recibido. A mi hermana, porque tú también has contribuido e influenciado en mi manera de ser, y a llegar hasta aquí.

En segundo lugar, a mi tutor Jesús, por haberme dado la oportunidad de realizar este proyecto, y compartir sus conocimientos sobre metodologías ágiles.

No me olvido de los compañeros que he conocido durante todos estos años, con tantos momentos compartidos, buenos y malos. Mención especial para ti, Adri, que juntos hemos conseguido llegar al final del camino, aquí tienes un amigo para siempre.

Por último, a la persona más especial en mi vida. Raquel, gracias por mostrar tu apoyo en todo momento, sobre todo en los tiempos difíciles, y por la paciencia que has tenido los últimos meses, en los que no hemos podido pasar mucho tiempo juntos.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

1 INTRODUCCIÓN Y OBJETIVOS

En este capítulo, se recoge una introducción del trabajo de fin de grado así como los objetivos que se persiguen con la realización del mismo. Además, se incluirá una breve descripción de cada uno de los capítulos incluidos para proporcionar una rápida información sobre el conjunto del documento.

1.1 Introducción

Es sabido que el mundo de las tecnologías de la información está en constante evolución. Esto implica la necesidad de permanecer en constante formación con el fin de mantenerse actualizado.

Por ello, el trabajo de fin de grado que nos ocupa, no trata solo del desarrollo de un producto, sino de establecer unas bases sólidas que sirvan para crecer profesionalmente en este sector tan cambiante.

El presente proyecto, además del desarrollo de un web site donde los usuarios puedan anunciar objetos de los que quieran desprenderse, tendrá mucha carga en la metodología a seguir para el propio desarrollo.

Si la metodología elegida no se adapta a las necesidades del proyecto, nos conducirá a una mala experiencia, y en ocasiones al fracaso del producto.

Por este motivo, además de hacer un rápido repaso a las tecnologías actuales a nivel de Ingeniería de Software, y ver las tendencias de cara al futuro, a lo largo del trabajo de fin de grado se realizará una comparativa entre las metodologías de software tradicionales (modelo en cascada) y las metodologías ágiles (SCRUM) para buscar aquella que se ajusta a lo que necesitamos.

1.2 Objetivos

Como decíamos en el apartado anterior, nuestro sector permanece en constante movimiento, y no dejan de salir novedades y nuevas necesidades. Por este motivo, trataremos de alejarnos del habitual esquema de desarrollo para estar preparados a lo que nos encontraremos al finalizar el trabajo de fin de grado.

El objetivo principal del proyecto, será desarrollar un portal web que permita a los usuarios publicar anuncios de objetos que deseen vender, y que al mismo tiempo ponga en contacto al usuario interesado en una oferta con el propietario de la misma.

Para completar el objetivo anterior, se deberá evaluar y comprender la situación actual de las aplicaciones web, y ver cuáles son las tendencias futuras de cara a desarrollar la aplicación para que se pueda adaptar a las novedades sin mayores dificultades.

Debido a la velocidad a la que aparecen nuevas tecnologías y características, un proyecto puede verse afectado y quedar obsoleto si no está preparado para el cambio. Por ello, el siguiente objetivo será estudiar las diferentes metodologías de desarrollo de software, y emplear una metodología que proporcione esa flexibilidad necesaria.

Por último, se deberán utilizar algunas herramientas o mecanismos que aporten valor añadido al propio desarrollo más allá de la propia funcionalidad requerida, como por ejemplo, realización de pruebas automáticas, simplificación del código para facilitar las tareas de mantenimiento, etc.

1.3 Fases del desarrollo

Antes de comenzar con el propio desarrollo, se realiza un estudio de la evolución de las tecnologías utilizadas en el desarrollo de aplicaciones Web para comprender en qué situación nos encontramos y el camino que está por venir, para establecer las bases de cara a una futura adaptación.

Después, seleccionaremos una metodología de desarrollo para estructurar y planificar el trabajo que ha de realizarse para hacer el trabajo de fin de grado de una manera ordenada.

Posteriormente, se procederá a obtener el material hardware y software necesario para poder desarrollar la aplicación que nos ocupa. Trataremos de reducir costes empleando herramientas de código abierto.

Más tarde, se estudiarán las necesidades y funcionalidades necesarias para que el portal web sea considerado como un producto que algún día pueda ponerse en un servidor real para que sea usado por cualquier usuario.

A partir de estas necesidades, se generarán los diferentes backlogs que se realizarán de manera iterativa en varios sprints de la misma duración, y así poder llevar un control periódico del estado real del desarrollo del portal, al tiempo que se va creando la documentación necesaria.

Por último, se procederá a completar la documentación con aquello que se considere oportuno para generar la memoria del trabajo de fin de grado, de modo que se pueda entender el trabajo realizado a lo largo del mismo.

1.4 Medios empleados

En este apartado se mencionarán los distintos elementos, tanto hardware como software, que han sido utilizados para el desarrollo del trabajo de fin de grado.

1.4.1 Hardware

El material hardware mínimo necesario para una aplicación web, es un **servidor de aplicaciones** en el que se aloje la propia aplicación, y un equipo en el que resida la **base de datos** que almacena la información de la misma.

Puesto que por ahora la aplicación no tendrá tráfico real, será suficiente con un ordenador personal para ocupar ambos roles.

1.4.2 Software

A continuación, se enumerarán los distintos programas utilizados para el desarrollo del trabajo de fin de grado:

- **Eclipse:** Entorno de programación que permite al usuario desarrollar tanto la parte frontend como backend de una aplicación web.

- **MySQL Workbench:** Herramienta para el diseño y la gestión de la base de datos.

- **Dropbox:** Almacenamiento de ficheros en la nube. Utilizado como repositorio para almacenar copias de seguridad de la aplicación, y otros ficheros pertenecientes al conjunto de este proyecto.

- **Microsoft Office 2010:** Empleado para la generación de este documento.

- **Navegadores:** Firefox, Google Chrome e Internet Explorer para realizar pruebas de modo que podamos proporcionar un producto compatible.

1.5 Estructura de la memoria

A continuación se explicará brevemente el contenido de cada uno de los capítulos que componen este documento, con el objetivo de proporcionar al lector una visión general del mismo.

- **Capítulo 1. Introducción y objetivos**

Se recoge una visión global del trabajo de fin de grado, comenzando con una breve descripción del trabajo realizado y los objetivos a alcanzar. Además, se mencionarán las distintas fases que han tenido lugar en el desarrollo del proyecto, así como los recursos empleados.

- **Capítulo 2. Estado de la cuestión**

En este capítulo, se realiza un estudio de la situación actual de las aplicaciones web y del camino que están tomando de cara al futuro, con vistas a adaptarse a las tecnologías emergentes.

Por otro lado, se exponen las características de las metodologías de desarrollo de software tradicionales y las de las metodologías ágiles.

- **Capítulo 3. Tecnologías utilizadas**

Se describen las tecnologías utilizadas en el proceso de desarrollo de la aplicación web, y la función que cumplen.

- **Capítulo 4. Sprint 0**

En el cuarto capítulo, se habla sobre la primera fase de desarrollo de la metodología de desarrollo escogida. Se explica cómo debe realizarse, cuál es el objetivo de esta fase y quiénes son los roles necesarios para que esta primera fase sea realmente productiva y contribuya al éxito final del producto.

- **Capítulo 5. Sprint 1-4**

En este capítulo, se recogen las historias de usuario realizadas en cada uno de los sprints, y se explicará el criterio elegido para establecer las prioridades de cada historia de usuario.

- **Capítulo 6. Conclusiones y trabajos futuros**

En el sexto capítulo, se expondrán las conclusiones alcanzadas tras la realización del trabajo de fin de grado, así como los posibles trabajos futuros que puedan realizarse partiendo del estado actual de la aplicación web desarrollada.

- **Capítulo 7. Gestión del proyecto**

En el último capítulo, se describirán los temas relativos a la planificación del trabajo y al tiempo empleado en desarrollo del mismo, así como su presupuesto.

- **Glosario**

En este apartado, se muestra un listado en orden alfabético de términos técnicos o poco conocidos, para que el lector pueda acudir en caso de no conocer el significado de alguna palabra contenida en este documento.

- **Bibliografía**

Se enumeran los enlaces de referencia utilizados a lo largo del desarrollo del trabajo de fin de grado. Se ordenarán según su aparición en el documento.

- **Anexo I**

En el último apartado, se incluirán los manuales y capturas de pantalla de los programas empleados, para que el lector pueda seguirlos en caso de estar interesado en el uso de los mismos.

2 ESTADO DE LA CUESTIÓN

En este capítulo, se realiza un estudio de la evolución de la Web en los últimos años, observando la situación actual de las mismas y las tendencias de futuro.

Puesto que cada vez hay más novedades en cuanto las tecnologías disponibles, es necesario tener esto en cuenta a la hora de comenzar un nuevo proyecto. Por ello, se repasarán las diferentes alternativas que nos ofrecen las metodologías de desarrollo software, con el objetivo de escoger una que nos permita cierta flexibilidad a la hora de adaptarse a los cambios.

2.1 Evolución de la Web

En este apartado, hablaremos brevemente del camino que ha seguido la Web hasta llegar a la situación actual, así como de las tendencias que marcarán el futuro próximo de la misma.

2.1.1 Camino previo

Como sabemos, internet supuso abrir las ventanas de la información al mundo, y consiguió superar las barreras de la distancia. Uno de los métodos que surgieron para difundir dicha información, fueron las páginas web.

Las primeras páginas, conocidas como **páginas estáticas**, tenían como objetivo proporcionar información al lector. Introducías una palabra en un motor de búsqueda y te mostraba una página con información sobre dicho término, nada más.

Posteriormente, se agregaron los **hiperenlaces** a dichas páginas estáticas, lo que permitió la navegación entre páginas estáticas sin necesidad de recurrir al buscador. Esto supone la base de lo que conocemos como World Wide Web.

2.1.2 Estado actual

Con la **Web 2.0**, se abrió un mundo de posibilidades prácticamente infinitas. La capacidad de permitir al usuario interactuar con la información mostrada mediante eventos lanzados al servidor, supuso un cambio de concepto. Pasó de ser un flujo unidireccional, donde la información partía del servidor para finalizar en la vista, a ser bidireccional, donde además, mediante los eventos lanzados por el usuario, la información viaja hacia el servidor (1).

Esto transforma los sitios web, dejando de ser exclusivamente fuentes de información, para ser una plataforma de comunicación, en la que la información va creciendo a medida que pasa el tiempo con la interacción de los usuarios.

El mayor ejemplo de este escenario es la Wikipedia (2). Es una página web, cuyo origen y mantenimiento de la información la realizan los propios usuarios, aunque es sabido que también tiene su lado negativo ya que la información puede no ser totalmente fiable.

Otro ejemplo claro de plataforma web son las redes sociales, que no dejan de ser páginas web. Los desarrolladores crean una página con la funcionalidad deseada, un diseño con menor o mayor complejidad según gustos, y proporcionan al usuario los mecanismos de interacción para que sean ellos mismos los que generen el contenido y hagan crecer el sitio web.

Por tanto, salta a la vista que se ha de cambiar la manera de afrontar un nuevo proyecto. Las páginas web ya no se crean exclusivamente pensando en un propósito concreto, sino que además están orientadas a un público objetivo, ya sea por rango de edad, como por gustos y preferencias.

Por estos motivos, en el desarrollo de nuevas páginas web se deben estudiar varios aspectos si se quiere lograr el éxito, y no desaparecer al poco tiempo por no ser competitivo en el mercado.

2.1.3 Tendencias de cara al futuro

En los últimos años, es muy difícil ver una empresa sin su propia página web. Por este motivo, cuando un usuario realiza una búsqueda en internet, y lo que encuentra son varias empresas del sector, hay que conseguir tener un distintivo que llame la atención e incline la balanza.

En empresas pequeñas, la creatividad y originalidad de los empresarios y los desarrolladores, puede llegar a ser suficiente para obtener un número aceptable de clientes, y sacar partido a la página web.

Sin embargo, para las grandes compañías no es suficiente con tener una página web llamativa, sino que tiene que estar dotada de otros aspectos para captar clientes potenciales, y por supuesto, evitar la fuga de clientes a la competencia directa.

Por ello, en los próximos apartados se hablará de los aspectos en los que se está poniendo mayor interés de cara al futuro.

Marketing y expertos del sector

Por los motivos expuestos en los párrafos anteriores, el web site de una empresa es un servicio más de la compañía, y por tanto, hay que poner los medios necesarios para que así sea.

El propietario de una empresa, o los directivos y socios de la misma, suelen tener el conocimiento de lo que sus clientes quieren. Sin embargo, puede haber casos en el que los perfiles de los clientes que utilizan internet, en vez de los contactos tradicionales, no coincidan.

Por ello, año a año se hace más notable la influencia de otros sectores a la hora de crear y mantener una página web. Las empresas con grandes recursos, recurren a especialistas en marketing y diseño, para conseguir una página bonita que agrade al usuario, pero que al mismo tiempo refleje de manera fiel la imagen de la propia empresa.

Esto está provocando un cambio en el diseño que se solía ver navegando por internet. Cada vez más, vemos como la página de inicio sirve de escaparate de la propia empresa, y el departamento de marketing es el encargado de situar un eslogan para captar clientes, o un conjunto de imágenes que transmitan dicho mensaje (3).

Además del aspecto visual, es necesario que la página cubra todas las necesidades que pueda tener un usuario. De ahí, que las grandes empresas, como por ejemplo las del sector bancario, contraten a expertos en banca que se dedican a examinar la web, con el objetivo de detectar los puntos débiles de la misma, o incluso mejorar la redacción técnica de la información que se muestra en pantalla.

Concepto multiplataforma

Otro aspecto a tener en cuenta, es cuándo y cómo acceden los usuarios a internet. Con la aparición de los smartphones y las últimas campañas de las compañías de telefonía móvil, el acceso a internet no está limitado a las proximidades del hogar, y se puede acceder desde teléfonos y tablets en el transporte público de camino al trabajo.

Esto hace que se modifique la manera de navegar (4). Se visita un mayor número de web sites, sin embargo se invierte menos tiempo en cada una de ellas. De ahí, como comentamos en el apartado anterior, la importancia del trabajo que realizan los diseñadores y especialistas en marketing, que deben ser capaces de llamar la atención del usuario.

Por estos motivos, se ha ido extendiendo el uso del "responsive design" (5) para permitir que una web sea compatible con los distintos dispositivos que proporcionan acceso a internet.

Single Page Interface (SPI)

Se trata de un concepto diferente de entender la Web, que ya se está empleando, y que en los próximos años será difícil encontrar un web site que no esté construido basado en SPI (6).

Con SPI, se sustituye la habitual navegación entre páginas de un web site, por el cambio de estado en una misma página. Los elementos que intervienen en el SPI son:

- **Modelo:** Datos que se muestran en pantalla, estructurados en XML o JSON para facilitar la transferencia de los datos con el servidor.

- **Plantillas:** Fragmentos de código HTML.

- **Eventos:** Acción que desencadena un cambio de estado.

El proceso, comienza por utilizar un elemento en el servidor conocido como “listener”, que se mantiene “escuchando” a la espera de recibir eventos por parte de la vista. Al recibir un **evento** en el estado A, se realizan las modificaciones oportunas en el **modelo** de datos, y se vuelve a enviar el modelo a la **vista**, junto con la indicación de que se debe realizar una transición al estado B.

Con esa información, la vista accede al DOM del documento para eliminar los elementos que no son necesarios, y añadir los nuevos, que se construyen a partir de las **plantillas** asociadas al estado B, mostrando los datos del modelo recibido del servidor.

Accesibilidad

Con el objetivo de llegar a todo el público posible, se está empezando a desarrollar cuidando varios aspectos, y uno de ellos es la accesibilidad (7).

Poco a poco, se están asentando unas bases de desarrollo y diseño, para que personas con discapacidades visuales puedan acceder al contenido sin problemas, ya sea evitando el uso de colores conflictivos, o bien, estructurando el contenido para que sea compatible con los lectores de páginas que utilizan las personas invidentes.

2.2 Metodologías de desarrollo de software

En este apartado, se hablará sobre algunas de las metodologías que se utilizan en el mundo laboral, tanto las tradicionales, como las que se empiezan a utilizar cada vez más en los últimos años.

Una metodología de desarrollo, consiste en un conjunto de pautas y procedimientos que sirven para ayudar a desarrollar un proyecto de manera organizada, para alcanzar un objetivo final.

Sin embargo, no existe una metodología que funcione de manera universal. De hecho, es habitual concebir las metodologías como 'marcos' metodológicos que son necesarios ajustar para cada organización y tipo de proyecto (8). Por ello, es muy importante conocer las alternativas disponibles, sus ventajas y desventajas, para escoger aquella que mejor se adapte al tipo producto que se quiere desarrollar.

Independientemente del tipo de metodología elegida, en los desarrollos de software hay unas fases o tareas que hay que realizar para generar un producto completo y de calidad (9). A continuación, se enumeran dichas fases con una breve definición de su propósito:

- **Análisis:** Se estudia el problema a resolver y si es viable afrontar el problema.
- **Especificación:** Se detallan los requisitos a cumplir por el producto generado, y las tareas a realizar. Suele estar integrada en la fase de análisis.
- **Diseño:** Se explica el cómo se va a llevar a cabo la resolución del problema analizado en la fase de análisis.
- **Implementación:** Periodo en el que se desarrolla la solución establecida en la fase de diseño.
- **Pruebas:** Fase de verificación del producto desarrollado.
- **Documentación:** Generación de la documentación oportuna, como manuales de usuario.
- **Mantenimiento:** Periodo en el que se corrigen posibles problemas detectados, una vez que el producto se entrega al cliente final.

2.2.1 Metodologías tradicionales

Este tipo de metodologías, en las que se han basado infinidad de proyectos, se basan en la definición de objetivos, requisitos y tareas. Todas y cada una de las fases mencionadas en el apartado anterior, deben estás definidas y registradas en los documentos pertinentes, para acudir en caso de necesidad.

Este hábito de realizar una documentación exhaustiva, donde cada fase es la entrada de la siguiente, guiada por la documentación, permite que todos los conceptos estén claros desde el inicio del mismo. Sin embargo, esto hace que los costes previstos al inicio se disparen ante cambios o problemas imprevistos, puesto que se basan en un **modelo predictivo** y no adaptativo.

Además, en esta documentación previa, también quedan definidos los distintos procesos que se realizarán a lo largo del proyecto, incluyendo las estimaciones en tiempo y coste.

Ciclo de vida

El ciclo de vida, es el conjunto de fases que forman un desarrollo y la manera en la que se suceden unos a otros. El más representativo y extendido en las metodologías tradicionales, es el conocido **ciclo en cascada**.

Este ciclo de vida, se caracteriza por establecer un orden secuencial de las fases de desarrollo que vimos en apartados anteriores. Esto implica que hasta que no se dé por finalizada una fase, no se puede comenzar con la siguiente. A continuación, mostramos un esquema ilustrativo antes de pasar a ver las ventajas y desventajas del mismo.

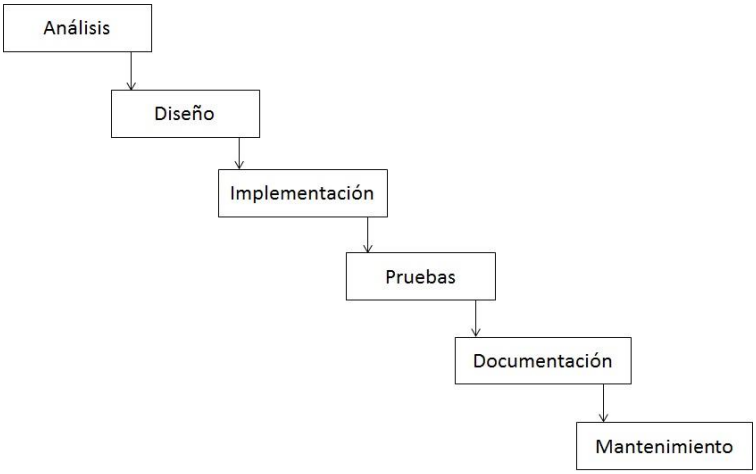

Ilustración 1. Modelo en cascada

Otro ciclo de vida empleado en estas metodologías, es el **modelo en V**. Este ciclo de vida, es muy similar al visto en párrafos anteriores, puesto que sigue una estructura lineal y secuencial. En la parte izquierda, se muestra la fase de análisis y especificación, y en la derecha la parte de pruebas, verificación e integración:

Ilustración 2. Modelo en V

Ventajas

- Todo está estructurado y definido, por lo que no se mezclan conceptos.
- Es perfecto para proyectos rígidos y bien definidos, similares a otros realizados con anterioridad.
- Tanto el cliente como los desarrolladores conocen las distintas fases que componen el proyecto.

Desventajas

- Requiere mucho tiempo completar todo el proceso.
- Las revisiones del producto son de gran complejidad al realizarse en las últimas fases.
- Cuando se detecta un fallo en la fase de pruebas, se genera un incremento de costes y de tiempo real, al tener que retroceder a fases previas para corregirlo.
- Los requisitos especificados en la fase de análisis son cerrados, y por tanto no se admite la creación de nuevos requisitos en fases posteriores del proyecto.
- El cliente no participa, y por tanto no conoce el resultado hasta el final.

2.2.2 Metodologías ágiles

La otra gran vertiente de las metodologías de desarrollo, son las conocidas como metodologías ágiles, y tiene a SCRUM como representante más importante. Estas metodologías, se utilizan cada vez más a medida que las empresas pierden el miedo al cambio. Se caracterizan por disminuir la cantidad de documentación generada, para dedicar más tiempo al propio desarrollo y generar código funcional en un tiempo relativamente corto, sin menospreciar la documentación, y basándose en el pensamiento Lean, donde se elimina todo aquello que no aporte valor al producto (10).

Estas metodologías surgieron en la década de los 90, como método alternativo a las metodologías tradicionales, que para muchos suponía un proceso burocrático y lento, que desviaba la atención del propio desarrollo.

Esto no significa que se deje totalmente de lado la organización del proyecto, sino que es un tema de reestructuración de las prioridades, quitando peso a ciertas tareas para favorecer otras que aporten valor real y demostrable.

Manifiesto ágil

Varios años después de la aparición de estas metodologías, un grupo de expertos organizaron una reunión para debatir sobre estas. De esa reunión, se extrajeron cuatro puntos en contraposición de las metodologías tradicionales (11):

Metodologías ágiles	Metodologías tradicionales
Individuos y su interacción	Procesos y herramientas
Software que funciona	Documentación exhaustiva
Colaboración con el cliente	Negociación contractual
Respuesta al cambio	Seguimiento de un plan

Tabla 1. Metodologías ágiles vs Metodologías tradicionales

Tras estos puntos, se desarrollan 12 principios que hoy en día se consideran como los principios básicos del movimiento ágil. A continuación, citamos el listado de dichos principios:

1. *Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.*
2. *Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.*

3. *Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los periodos breves.*
4. *Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.*
5. *Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.*
6. *La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.*
7. *El software que funciona es la principal medida del progreso.*
8. *Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.*
9. *La atención continua a la excelencia técnica enaltece la agilidad.*
10. *La simplicidad como arte de maximizar la cantidad de trabajo que hace, es esencial.*
11. *Las mejores arquitecturas, requisitos y diseños emergen de equipos que se auto organizan.*
12. *En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia.*

Modelo iterativo e incremental

Estos principios surgieron, entre otras cosas, como medida para evitar los puntos débiles del modelo en cascada usado en las metodologías tradicionales.

Como vemos, estos principios transmiten una idea común, un pensamiento: la **proximidad**. Al evitar largos periodos de planificaciones y análisis, se consigue tener un control mayor sobre las tareas a realizar, y mantiene a todas las personas involucradas en el proyecto (12). En el siguiente esquema, veremos las fases intervinientes en este modelo de desarrollo:

Ilustración 3. Modelo iterativo e incremental

Como se puede apreciar, este tipo de modelo consiste en ir haciendo entregas parciales del producto, de modo que el cliente tenga disponible una versión temprana del producto funcionando. Tras cada entrega, el cliente valida las funcionalidades entregadas, informa de los posibles errores y establece las tareas a realizar en la próxima iteración.

Las tareas que se desarrollan en cada iteración, pueden ser nuevas funcionalidades, correcciones de funcionalidades ya entregadas, o mejoras de las mismas. De esta manera, se reduce de manera importante el riesgo del proyecto al tener un feedback constante del propio cliente, además de conseguir un producto que se ajusta completamente sus propias preferencias.

Roles en SCRUM

En esta metodología, existen unos roles bien definidos que trabajan como un único equipo a lo largo de todo el proyecto (13):

- **Scrum master:** Persona que lidera al equipo guiándolo para que cumpla las reglas y procesos de la metodología. Gestiona la reducción de impedimentos del proyecto y trabaja con el **Product Owner** para maximizar el ROI.
- **Product owner (PO):** Representante de lso accionistas y clientes que usan el software. Se focaliza en la parte de negocio y el es responsable del ROI del proyecto (entregar un valor superior al dinero invertido). Traslada la visión del proyecto al equipo, formaliza las prestaciones en **historias** a incorporar en el **Product Backlog** y las reprioriza de forma regular.
- **Team:** Grupo de profesionales con los conocimientos técnicos necesarios y que desarrollan el proyecto de manera conjunta llevando a cabo las **historias** a las que se comprometen al inicio de cada sprint.

Historias de usuario

En Scrum, las tareas a realizar en cada iteración se conocen como historias de usuario. En comparación con las metodologías tradicionales, vendrían a ser los requisitos software y las especificaciones de los requisitos (14).

El encargado de crear las historias de usuario, es el product owner (PO en adelante). En cada una de ellas, que estarán identificadas numéricamente, aparecerá una breve descripción, la condición de validación y la prioridad en negocio.

Gracias a estas historias de usuario, se consigue dividir el proyecto en pequeñas entregas funcionales, facilitando la estimación de las mismas, y fomentando la estrecha relación con el cliente a lo largo del desarrollo.

2.2.3 Justificación

Tras hacer una comparativa de los dos representantes más importantes de las dos vertientes de metodologías de desarrollo de software, en este apartado se justificará por qué hemos decidido seguir SCRUM para el desarrollo del trabajo de fin de grado.

Durante décadas, las empresas han utilizado las metodologías tradicionales con mayor o menor éxito. Sin embargo, en un sector como el de las tecnologías de la información, en muchas ocasiones las necesidades de los usuarios avanzan más rápidamente que el tiempo de desarrollo en este tipo de metodologías.

Cada vez son más las empresas que se animan, y empiezan a ver con buenos ojos el movimiento ágil. La nueva ingeniería del software, requiere de una participación más directa del cliente en aras de reducir el **time to market**. Al realizar pequeñas entregas, se tiene un mayor control sobre las funcionalidades desarrolladas, las carencias del producto, o detalles que al inicio del proyecto se pasaron por alto.

Gracias al **modelo iterativo e incremental**, se pueden incluir estos detalles de manera que conseguimos un producto de calidad, donde no solo cumple con el objetivo inicial, sino que lo hace de la mejor manera posible, cuidando la calidad del desarrollo, y obteniendo como resultado un producto final que se ajusta a las necesidades reales del cliente.

Por estos motivos, considero que la mejor manera de llevar a cabo el desarrollo de una aplicación empresarial, y ser lo más adaptativos posibles para reducir el time to market, es utilizar las metodologías ágiles, y tener esa flexibilidad necesaria ante los cambios que puedan producirse a lo largo de la vida del mismo.

3 TECNOLOGÍAS UTILIZADAS

En este capítulo, enunciaremos las tecnologías utilizadas para el desarrollo del trabajo de fin de grado. Su uso está muy extendido puesto que todas ellas son de fácil acceso al ser tecnologías de código abierto.

3.1 HTML5

Se trata de la quinta versión del lenguaje básico de la World Wide Web, el HTML. El desarrollo de este lenguaje marcado es regulado por W3C, al igual que sus versiones anteriores. En esta nueva versión, aparecen nuevas etiquetas como por ejemplo <article>, <header> y <footer>, que ayudan al programador a maquetar la página web.

Esta tecnología es soportada por las últimas versiones de los diferentes navegadores (Internet Explorer, Google Chrome y Firefox, entre otros). En nuestro caso, utilizaremos el código HTML5 en los ficheros de extensión JSP, que permite el uso de código Java en páginas HTML.

3.2 JQuery

Es una biblioteca de Javascript, y por tanto se ejecuta en el lado del cliente (navegador), que simplifica la manera en la que podemos interactuar con el DOM, y cambiar los elementos HTML dinámicamente. En nuestro caso, se ha utilizado para modificar elementos a la hora de validar formularios.

3.3 CSS

Son hojas de estilo en cascada, usadas para definir el estilo y formato de los documentos basados en etiquetas. En nuestro caso, se han utilizado para la maquetación de los ficheros que contienen los ficheros HTML.

3.4 JPA

Se trata de un framework de Java, que permite manejar datos relacionales en aplicaciones Java. Esto se realiza mediante elementos conocidos como entidades. Cada entidad, que consiste en una clase Java, representa una tabla de la base de datos. En cada entidad, tendremos tantos atributos como columnas tenga la tabla que representa.

Gracias a ello, conseguimos preservar las ventajas de la programación orientada a objetos tras acceder a la base de datos.

3.5 MySQL

Es un sistema de gestión de base de datos, que además es compatible con multitud de interfaces de programación de distintos lenguajes. En nuestro caso, realizaremos los accesos a la base de datos desde el código Java, ayudándonos de las bondades de JPA, utilizando lenguaje SQL. La base de datos de nuestro web site, almacenará los datos de los usuarios y de las ofertas creadas en él.

3.6 JUnit

Es un conjunto de bibliotecas, utilizadas para realizar pruebas unitarias en aplicaciones Java. Permite la ejecución controlada de clases Java, sin necesidad de iniciar el servidor de aplicaciones. Para probar un método, se establecen los parámetros de entrada y la salida esperada, y tras ejecutar el test, podemos comprobar si el método está bien implementado.

Además, también sirve para realizar pruebas de regresión cuando se han realizado modificaciones sobre el código, y poder verificar que el funcionamiento de todos los métodos sigue siendo el esperado.

3.7 Mockito

Es un framework para Java, cuyo objetivo es contribuir en la realización de las pruebas unitarias. En ocasiones, queremos probar un método que contiene elementos que no podemos definir en el propio test. Por ello, gracias a mockito, podemos simular comportamientos y definir qué sucederá al ejecutar una acción contenida en el método que queremos probar.

3.8 Log4j

Se trata de una biblioteca desarrollada por Apache, que permite a los desarrolladores elegir los niveles y la granularidad de los ‘logs’ que se muestran durante la ejecución de una aplicación.

En nuestro caso, utilizaremos esta biblioteca para configurar unos ficheros de log, en los que se almacenen trazas de distintos niveles, para guardar un registro de logs y poder utilizarlos a la hora de depurar el código.

4 SPRINT 0

En este capítulo, se recogerán las tareas realizadas en la primera fase conocida como Sprint 0 (o 'zero'). Estas tareas, forman parte de la planificación inicial del desarrollo del producto.

Esta fase, suele durar entre una semana y un mes, dependiendo de varios factores: complejidad del producto, formación del equipo, obtención del material y estabilización del entorno del trabajo.

4.1 Formación del equipo

La primera tarea, consiste en terminar de contratar al personal que va a participar el proyecto, si todavía queda algún puesto que cubrir. Además de las personas físicas, es la fase ideal en la que se ha de reunir el material necesario para realizar el trabajo sin demoras. Si no se realiza en esta fase, implica que el comienzo del desarrollo se vaya atrasando, con lo que perdemos productividad.

Por tanto, además de conseguir el hardware necesario, deberán prepararse los entornos para solucionar los problemas que puedan surgir durante la instalación e integración del mismo, y conseguir estabilizarlo con vistas a comenzar el desarrollo en cuanto haya tareas suficientes para comenzar un sprint.

Por último, se utilizará esta fase para iniciar al equipo en la metodología de trabajo a seguir, o repasar conceptos en caso de que todo el equipo de trabajo lo conozca, para trabajar todos en la misma dirección y no perder tiempo de desarrollo en otras cuestiones.

Un ejemplo de este tipo de iniciativas para repasar conceptos, es el **Agile Inception** (15). Consiste en realizar un conjunto de actividades grupales, en el que se fomenta la creatividad y la reflexión de los participantes, afrontando diferentes problemas propuestos en cada uno de los ejercicios.

4.2 Requisitos no funcionales

En este apartado, se mostrarán los requisitos no funcionales establecido para nuestro web site. Los datos que se mostrarán en cada uno de ellos son:

- **Identificador:** Será único para cada requisito, y seguirá la nomenclatura "RNF-XX", donde XX será un número entero.

- **Nombre:** Sirve para ayudar a la identificación del requisito, sin necesidad de leer el resto de campos.

- **Descripción:** Explicación del requisito no funcional, de manera clara y concisa.

- **Prioridad:** Prioridad del requisito, para establecer una escala objetiva (Alta, media, baja).

- **Estabilidad:** Tendrá tres posibles valores:
    - Alta: No se modificará bajo ninguna circunstancia
    - Media: Es posible que sufra alguna modificación durante la vida del producto.
    - Baja: Se modificará frecuentemente durante la vida del producto.

Identificador: RNF-1	**Nombre: Tiempo de respuesta**
Descripción: En ningún caso, el tiempo máximo de respuesta será de 1 segundo.	
Prioridad: Alta	
Estabilidad: Alta	

Tabla 2. RNF-1. Tiempo de respuesta

Identificador: RNF-2	**Nombre: Usuarios concurrentes**
Descripción: Debe proporcionar servicio de manera concurrente al menos a 2 usuarios.	
Prioridad: Alta	
Estabilidad: Media	

Tabla 3. RNF-2. Usuarios concurrentes

Identificador: RNF-3	Nombre: Usabilidad
Descripción: La navegación debe ser intuitiva y no tener estados huérfanos, sin retorno.	
Prioridad: Alta	
Estabilidad: Alta	

Tabla 4. RNF-3. Usabilidad

Identificador: RNF-4	Nombre: Compatibilidad navegadores
Descripción: El web site deberá ser compatible con las últimas versiones de los navegadores Google Chrome, Firefox e Internet Explorer.	
Prioridad: Alta	
Estabilidad: Alta	

Tabla 5. RNF-4. Compatibilidad navegadores

Identificador: RNF-5	Nombre: Contenido
Descripción: La información mostrada debe ser clara, para que no lleve a confusiones al usuario.	
Prioridad: Alta	
Estabilidad: Alta	

Tabla 6. RNF-6. Contenido

Identificador: RNF-6	Nombre: Crecimiento orgánico
Descripción: El web site estará estructurado en paquetes, para contribuir al crecimiento orgánico de la aplicación al añadir nuevas funcionalidades.	
Prioridad: Alta	
Estabilidad: Alta	

Tabla 7. RNF-6. Crecimiento orgánico

4.3 Product backlog

Esta tarea es la más importante en esta primera fase. En ella, el cliente creará el product backlog, que contendrá la pila del producto con las funcionalidades a desarrollar.

Una de las ventajas de las metodologías ágiles, es que no es necesario analizar al inicio todas las funcionalidades que se desean incorporar al producto, sino que se pueden ir añadiendo en sprints futuros a medida que va creciendo el producto.

A continuación, mostramos el backlog del primer sprint, para mostrar la información registrada en el mismo:

	SPRINT	INICIO	DURACIÓN
	1	4-nov-13	21

PILA DEL SPRINT				
Backlog ID	Tarea	Tipo	Estado	Responsable
tiendaWeb	Crear cuenta usuario	Pruebas	Terminada	Álvaro Cedillo
	Consultar perfil usuario	Pruebas	Terminada	Álvaro Cedillo
	Modificar perfil usuario	Pruebas	Terminada	Álvaro Cedillo
	Eliminar cuenta usuario	Pruebas	Terminada	Álvaro Cedillo
	Confirmacion registro cuenta usuario	Pruebas	Terminada	Álvaro Cedillo
	Recuperar contraseña usuario	Pruebas	Terminada	Álvaro Cedillo
backofficeWeb	Crear cuenta administrador	Pruebas	Terminada	Álvaro Cedillo
	Consultar perfil administrador	Pruebas	Terminada	Álvaro Cedillo
	Modificar perfil administrador	Pruebas	Terminada	Álvaro Cedillo
	Eliminar cuenta administrador	Pruebas	Terminada	Álvaro Cedillo
	Confirmacion registro cuenta administrador	Pruebas	Terminada	Álvaro Cedillo
	Recuperar contraseña administrador	Pruebas	Terminada	Álvaro Cedillo

Ilustración 4. Backlog sprint 1

Como el desarrollo está terminado, todas las tareas aparecen con estado “Terminadas” y tipo “Pruebas. Por ello, en el siguiente párrafo enumeramos los posibles valores de ambas columnas.

En la columna **tipo**, las tareas pueden tener lo valores: “Reunión”, “Análisis”, “Diseño”, “Codificación” y “Pruebas”. En cuanto a la columna **estado**, los posibles valores son: “Pendiente”, “En curso”, “Terminada” y “Eliminada”.

4.3.1 Historias de usuario

En el sprint 0, se define la manera en la que se van a recoger las distintas historias de usuario a lo largo del proyecto para conservar la homogeneidad. Además, se comenzarán a definir algunas historias de usuario hasta conseguir un volumen suficiente como para empezar el primer sprint.

Como excepción, para no duplicar tablas en este documento, no mostraremos ninguna historia de usuario en este capítulo, sino que se listarán en el sprint correspondiente del capítulo 4. A continuación, mostramos una tabla de ejemplo y definiremos el objetivo de cada uno de los campos.

Número:	**Nombre:**
Prioridad:	
Descripción:	
Criterio de aceptación:	

Tabla 8. Formato de historia de usuario

- **Número:** Identificador único de la historia de usuario. Será un número entero de 1 a N, siendo N el número total de historias recogidas.

- **Nombre:** Sirve para ayudar a la identificación de la historia de usuario sin necesidad de leer el resto de la tabla.

- **Prioridad:** Escala numérica que marca la importancia de la historia de usuario respecto al resto.

- **Descripción:** Breve explicación de quién es el beneficiado por la historia de usuario, y el beneficio obtenido. Se formulará siempre de la misma manera, usando las tres palabras marcadas en negrita. Por ejemplo: "**Como** usuario, **quiero** poder registrarme en la web **para** poder acceder a la parte privada de la aplicación".

- **Criterio de aceptación:** Descripción que indica cuándo la historia de usuario se puede considerar completada.

Durante el proceso de desarrollo de las funcionalidades descritas en las historias de usuario, se realizará un seguimiento diario, conocido en Scrum como "**Daily**". Consiste en una reunión diaria de unos 15 minutos, donde cada participante debe responder a tres preguntas: ¿Qué hiciste ayer? ¿Qué tienes previstos hacer hoy? ¿Hay algún bloqueo que impida completarlo?

Gracias a esta reunión, todo el equipo tiene conocimiento del estado de las historias de usuario que se están tratando, al tiempo que se comunican posibles dependencias entre compañeros.

4.4 Arquitectura Web

Una vez que tenemos una visión general de las funcionalidades solicitadas por el cliente, es el momento de definir la arquitectura que tendrá nuestra página web. En este sentido, hay dos aspectos de los que debemos encargarnos: Arquitectura lógica y arquitectura física.

Arquitectura lógica

En esta parte de la arquitectura, se define cómo se relacionarán los elementos entre sí. En nuestro caso, hemos elegido una arquitectura muy extendida en Internet, como es la arquitectura MVC (Modelo-Vista-Controlador) (16). Adjuntamos una imagen que describe perfectamente las partes implicadas en esta arquitectura.

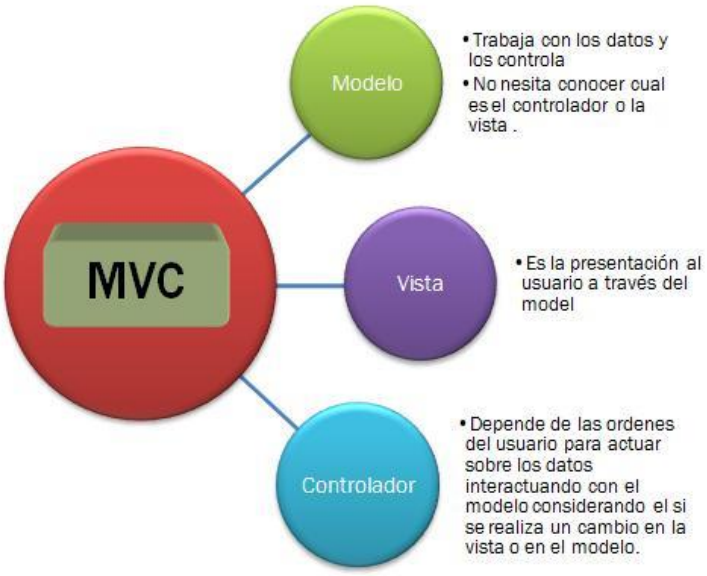

Ilustración 5. Arquitectura MVC

Su utilidad como patrón, permite separar los distintos elementos que forman una aplicación web, lo que permite desarrollar en paralelo y acortar tiempos de desarrollo. En nuestro caso, solo habrá un desarrollador, sin embargo creemos oportuno trabajar con este paradigma de comunicación entre elementos.

A continuación, mostraremos un esquema con los componentes lógicos que componen nuestra página web.

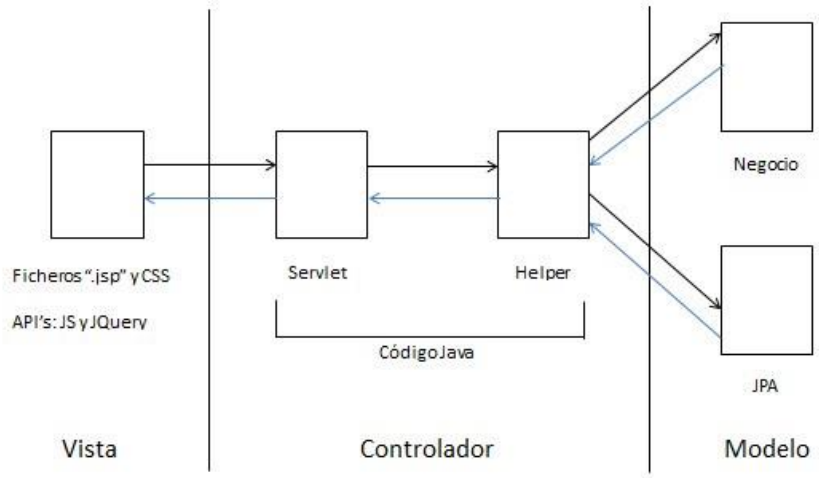

Ilustración 6. Estructura de la arquitectura lógica

Partiendo de este esquema, mostramos un grafo que describe el flujo de información que se realiza cuando el usuario interactúa con nuestro web site.

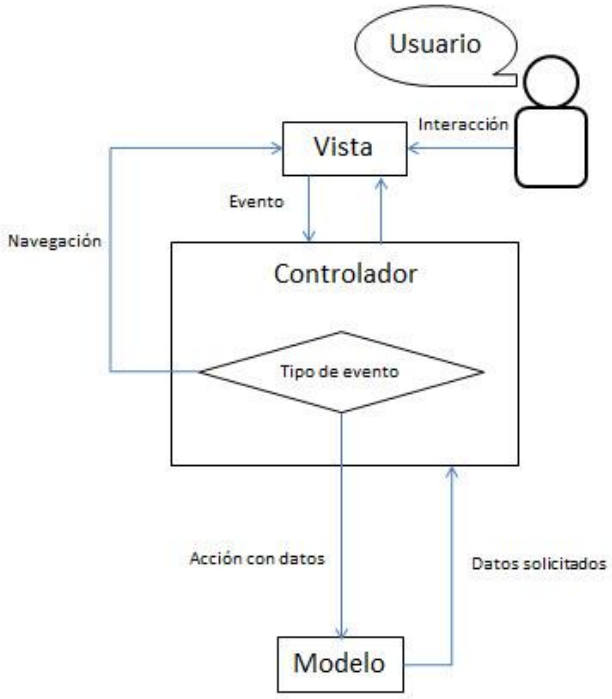

Ilustración 7. Flujo de información entre elementos

Arquitectura física

En cuanto al aspecto físico de la arquitectura, describe la situación física de los elementos que forman una aplicación web. Para desplegar una aplicación web, son necesarios varios elementos que definimos a continuación:

- **Base de datos:** Contiene los datos de la aplicación, tanto de los usuarios como de las ofertas.
- **Servidor:** Contiene el código generado en el desarrollo del portal web, y se encarga de desplegarlo para que sea accesible.

Una vez que conocemos los elementos necesarios, se debe realizar una valoración de la carga que tendrá nuestra página web, y ver cómo estructurar los elementos para dar soporte. En la actualidad, las páginas web que quieren tener éxito deben primar tres aspectos: Tiempo de respuesta, usabilidad y accesibilidad.

Por ello, para conservar un tiempo de respuesta adecuando, debemos tener en mente la **concurrencia de usuarios** que tendremos que soportar. Asumiendo que nuestra página tendrá dos usuarios concurrentes, nos podemos permitir alojar en un mismo equipo el servidor de aplicaciones y la base de datos sin que el **tiempo de respuesta** se vea afectado.

Si la concurrencia fuese mayor de 20 usuarios, dato establecido en la configuración del servidor como en la instancia de la base de datos, sería necesario utilizar dos servidores para conservar la alta disponibilidad del sistema, y se mantengan estables los tiempos de acceso. Esto último, implica la necesidad de incorporar un **balanceador** de carga que se encargue de distribuir las peticiones entre los servidores existentes.

Para que sea más claro, mostraremos una imagen donde la base de datos y el servidor se encuentran en equipos distintos, aunque recordamos que en nuestro caso se encuentran en el mismo equipo.

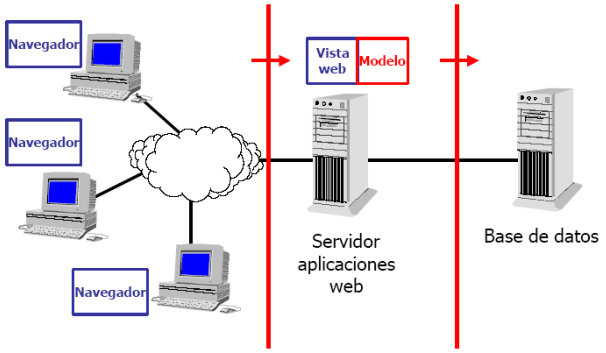

Ilustración 8. Estructura de la arquitectura física

Estructura de la web

Además de la propia arquitectura comentada en las dos secciones anteriores, es necesario definir en esta etapa la estructura web de nuestra aplicación.

El objetivo principal a la hora de establecer la estructura de un sitio web, es proporcionar al usuario un web site bien organizado, en el que la información sea clara y los caminos estén bien definidos, para que el proceso de navegación no le suponga un problema al usuario, es decir, cuidar la **usabilidad** de nuestra aplicación (17).

Esto último parece una obviedad, pero hoy en día, aún encontramos decenas de páginas en las que la navegación no es nada intuitiva, y acabamos en secciones de las que no sabemos salir de una manera natural, y terminamos por abandonar la página.

Para evitar estos problemas, utilizaremos una estructura web mixta, de modo que aunque existan varias secciones bien diferenciadas (ofertas, usuarios, preguntas frecuentes, etc.), con su propia jerarquía de niveles, permitamos al usuario navegar de una a otra de manera intuitiva.

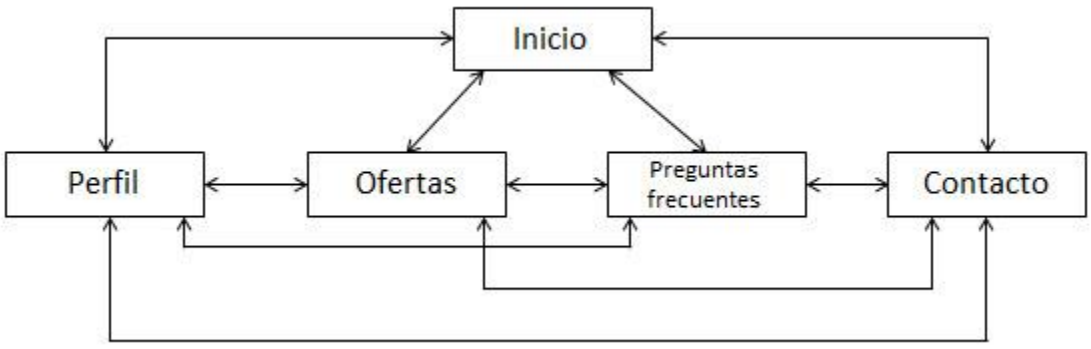

Ilustración 9. Estructura web

En la imagen superior, representamos el esquema web de nuestro proyecto. Gracias a él, proporcionamos al usuario total libertad a la hora de navegar, pudiendo cambiar de sección sin tener que volver al inicio de la aplicación, y sin dejar secciones huérfanas de las que no es posible volver.

4.5 Organización del código

Otra actividad recomendable en esta fase, es establecer desde el comienzo cómo vamos a organizar los ficheros que componen nuestra aplicación web.

Funcionalmente, es válido ir generando los ficheros en un mismo paquete, ya sean servlets, imágenes, estilos, etc. Sin embargo, a medida que la aplicación crece, se dificulta la tarea de localizar una funcionalidad.

Por tanto, es altamente recomendable reunir al equipo de desarrollo para establecer unas bases de organización de paquetes. En los siguientes párrafos, mostraremos la organización establecida para el desarrollo de nuestra página web.

Lo primero que hay que saber, es cuántos proyectos vamos a generar para completar nuestra aplicación. En nuestro caso, desarrollaremos dos proyectos independientes. Uno para la parte de la administración, a la que se accederá por intranet, y otra para los usuarios, que se accederá a través de internet.

¿Por qué hacerlo en proyectos separados? Puesto que no existen dependencias entre ambas aplicaciones, tenemos la posibilidad de separarlo. Con ello, conseguimos independencia y libertad para reiniciar las aplicaciones de manera independiente, sin afectar al otro en caso de que surjan problemas de seguridad, errores críticos o que haya que realizar tareas de mantenimiento.

Del mismo modo, en caso de utilizar alojamiento web externo, se podría decidir alojar cada uno de los proyectos en servidores de aplicaciones distintos, de manera que si alguno de los servidores se ve comprometido o sufre problemas ajenos a nuestra aplicación, no perdamos el acceso a ambos proyectos a la vez.

En cuanto a la organización de la paquetería de la que se compone cada una de las aplicaciones, será la misma por cuestión de homogeneidad, separando los ficheros frontend de los correspondientes a la parte backend.

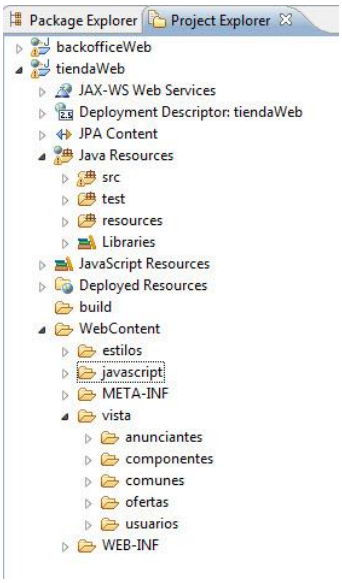

Ilustración 10. Estructura física del proyecto

En la parte frontend, contenida en la carpeta 'WebContent', tenemos tres carpetas independientes, para separar los estilos, los ficheros de javascript y los ficheros jsp (carpeta 'vista'). De esta manera, el desarrollador sabrá perfectamente en que carpeta deberá incluir un nuevo fichero en caso de necesitarlo.

En cuanto a la parte java, contenida en la carpeta 'Java Resources', adjuntamos otra imagen para ver la paquetería con más detalle.

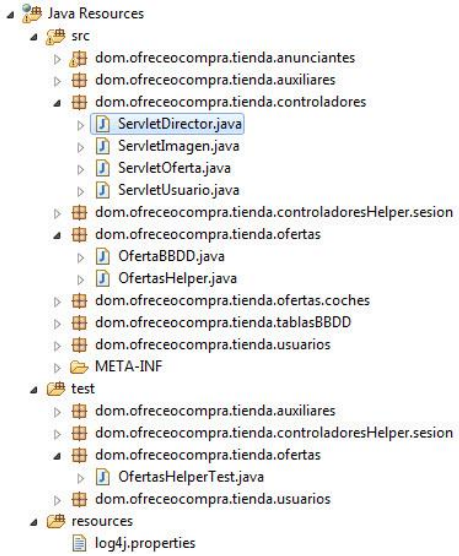

Ilustración 11. Organización por paquetes

En este caso, tenemos otros tres directorios. En la carpeta ‘resources’, se sitúan los ficheros de configuración que pudieran necesitarse en el código Java. En nuestro caso, solo tenemos el fichero de configuración de los logs.

En la carpeta ‘src’, nos encontramos con varios paquetes cuya nomenclatura tiene la misma raíz (‘*dom.ofreceocompra.tienda’*). Este tipo de nomenclatura, es una convención para que todos los desarrolladores utilicemos el mismo patrón y contribuir para establecer unas bases sólidas, y favorecer el **crecimiento orgánico** del producto de manera controlada.

Como se puede apreciar, cada uno de los paquetes está orientado a albergar distintos grupos de funcionalidades, y vemos por ejemplo, que las funcionalidades de las ofertas, están separadas de las funcionalidades relativas a los usuarios.

Por último, tenemos la carpeta ‘test’, que contiene los test unitarios de las clases Java incluidas en la carpeta ‘src’. Los paquetes que aparecen en este directorio, son los mismos que los de las clases Java a probar, de manera que los métodos de estas clases sean visibles por los test unitarios.

4.6 Valor añadido

Por último, se establecerán unas pautas que doten de calidad y valor añadido al producto que estamos desarrollando.

En muchas ocasiones, solo se tiene en cuenta el propio desarrollo de las funcionalidades solicitadas, olvidándose por completo de tareas que tarde o temprano hay que realizar, como es el mantenimiento del producto.

Por ello, es importante hacer ver al cliente estos detalles, que aunque no aporten nuevas funcionalidades, contribuirán de manera muy positiva a conseguir un producto mucho más controlado al tener un conjunto de mecanismos que faciliten tanto el mantenimiento del mismo, como futuras mejoras.

En los próximos apartados, se mostrarán las medidas adoptadas en el trabajo de fin de grado para dotar de valor añadido a la aplicación desarrollada.

Refactorización

La refactorización, es una técnica que puede costar entender y llevar a cabo al principio. Sin embargo, con un poco de práctica, los resultados comienzan a dar sus frutos (18). Es habitual que el desarrollo de un producto requiera miles de líneas de código. Como podemos imaginar, el mantenimiento de tantas líneas de código puede llegar a ser un dolo de cabeza.

Aquellas personas que en algún momento se hayan tenido que dedicar a depurar código, estarán de acuerdo en que la mayor parte del tiempo nos lo pasamos releyendo el código decenas de veces, debido a la complejidad del código o a la extensión del mismo.

Por ello, hay que hacer hincapié en la importancia de generar código de calidad que facilite este tipo de tareas, seamos o no los encargados de realizarlas. El primer motivo por el cual se produce esto, es que comenzamos a desarrollar como locos para conseguir la funcionalidad deseada, sin parar a pensar la mejor forma de llegar a ella.

Por tanto, lo primero que debemos hacer es un pequeño análisis de lo que queremos hacer para no realizar pasos innecesarios y conseguir así reducir el número de líneas. Es cierto que en ocasiones, el problema es tan complejo que es necesario comenzar con el desarrollo para ir aclarando el problema y allanar el camino a la solución.

La complejidad de una funcionalidad, no debe ser motivo para justificar la no utilización de esta técnica. Siempre es posible **reorganizar el código** en funciones más pequeñas para facilitar la comprensión del mismo.

Como dije en líneas anteriores, al principio es complicado dominar esta disciplina porque estamos demasiado acostumbrados a escribir líneas de código como locos, sin embargo debemos poner de nuestra parte para eliminar esos malos vicios y conseguir evolucionar como profesionales.

Por ello, aparte de analizar previamente la situación, es necesario repasar el código final generado para cumplir con esta tarea. El objetivo de la refactorización, es que el código sea lo más simple posible, y que cada función o método realice una sola tarea, para facilitar la depuración del mismo.

Test unitarios

Puesto que se realizan varias entregas a lo largo del desarrollo de un producto en las metodologías ágiles, puede suceder que al implementar nuevas funcionalidades dejen de funcionar adecuadamente.

Esto sucede en etapas avanzadas del desarrollo, a medida el aumenta la complejidad de la aplicación. Aunque este hecho no se pueda evitar, sí que se pueden emplear herramientas que nos permitan detectar el problema con suficiente margen como para solucionarlo antes de realizar la entrega correspondiente.

La solución más obvia, es realizar pruebas exhaustivas de toda la aplicación antes de la entrega, sin embargo, el lector estará de acuerdo que cuando el tamaño de la aplicación comienza a ser respetable, es muy difícil cubrir todas las posibles pruebas.

Por ello, hay que utilizar otros métodos que permitan la ejecución de estas pruebas en muy poco tiempo. La solución a este problema, es que los **test unitarios** lo hagan por nosotros (19).

Estos test, permiten la ejecución automática del código desarrollado, partiendo de unos parámetros que nosotros le proporcionamos al desarrollarlos. De este modo, si el resultado esperado en alguno de ellos no es correcto, el test correspondiente a dicha funcionalidad fallará.

A continuación, mostramos un ejemplo muy sencillo de un test unitario, que evalúa un método que genera cadenas de caracteres aleatorios.

AleatorioTest.java

```
1  package dom.ofreceocompra.tienda.auxiliares;
2
3  import org.junit.Assert;
6
7  public class AleatorioTest {
8      @Test
9      public void pruebaAleatorio() {
10         int longitud = 5;
11         String aleatorio = Aleatorio.generarAleatorio(longitud, "numeros");
12
13         Assert.assertTrue(aleatorio != null);
14         Assert.assertTrue(aleatorio.length() == longitud);
15     }
16
17     @Test
18     public void pruebaAleatorioAlfanumericoTest() {
19         int longitud = 5;
20         String aleatorio = Aleatorio.generarAleatorio(longitud, "");
21
22         Assert.assertTrue(aleatorio != null);
23         Assert.assertTrue(aleatorio.length() == longitud);
24     }
25 }
26
```

Ilustración 12. Ejemplo de test JUnit

Como vemos en la imagen, este test realiza dos pruebas sobre el método 'generarAleatorio' de la clase 'Aleatorio'.

- Cadena numérica con una longitud de 5 caracteres.
- Cadena alfanumérica con una longitud de 5 caracteres.

Al ejecutar el test, obtenemos la siguiente resultante a la ejecución del mismo. Con cada test, podemos obtener tres respuestas distintas:

- **Test correcto:** La salida obtenida coincide con la salida esperada.
- **Error:** Se ha producido un error durante la ejecución del test. Normalmente sucede por falta de datos previos para la correcta ejecución.
- **Failure:** La salida obtenida no coincide con la salida esperada.

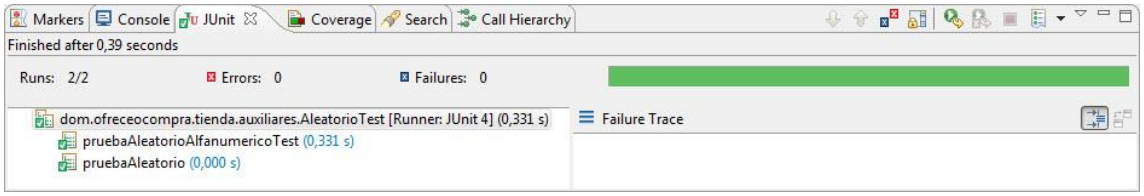

Ilustración 13. Resultado de ejecución de test unitario

Como vemos, la ejecución de nuestro test ha sido exitoso, y podemos concluir que la funcionalidad del método probado es adecuada.

Para contribuir a la creación de estos test, es necesario que los métodos que se quieran comprobar sean lo más concretos posibles, y que se puedan probar de manera independiente, de ahí el nombre de este tipo de test.

En los últimos años, ha surgido una manera diferente de afrontar el desarrollo de software utilizando los test unitarios. Este proceso es conocido como **TDD** (20), y parte de la premisa de comenzar el desarrollo creando los test unitarios, y posteriormente, el desarrollo de la funcionalidad. A continuación, mostramos el proceso con más detalle:

1. Se selecciona la funcionalidad a probar.
2. Se crea el test unitario.
3. Ejecutar el test → Falla, ya sea por error o por salida inesperada.
4. Crear código para resolver el test.
5. Ejecutar el test → Correcto, la salida es la esperada.
6. Refactorización del código.
7. Ejecutar el test → Correcto, la refactorización no ha afectado a la funcionalidad.

La mayor ventaja de este proceso, es que conseguimos que el código necesario para desarrolla una funcionalidad sea mínimo, además de contribuir a construir un código más legible, facilitando las tareas de mantenimiento del mismo.

Sistema de logs

Otro de los problemas que nos encontramos cuando aparecen errores, es la localización del origen y el motivo por el que falla. Una solución muy frecuente en el periodo docente, durante el desarrollo de las prácticas de diversas asignaturas, es imprimir datos en la consola de ejecución, para ver si pasar por un cierto punto y ver el valor de las variables implicadas.

Esta solución, puede ser válida durante el desarrollo de la propia funcionalidad, sin embargo su utilidad se desvanece completamente en los procesos de mantenimiento y detección de errores.

El objetivo de estos sistemas, es el mismo que la solución anterior, con la ventaja de que podemos almacenar en ficheros la información que se muestra en la consola en tiempo de ejecución.

De esta manera, podemos acudir en cualquier momento a esos ficheros, incluso días después de producirse el error, para observar las trazas almacenadas y comenzar con el estudio del problema causante del error.

En nuestro caso, hemos utilizado la librería de logs de Apache, conocida como **log4j**. Esta librería, permite establecer distintos niveles a las trazas generadas, con lo que podemos elegir qué trazas mostrar en los ficheros de log y cuáles no (21).

En esta ocasión, hemos utilizado solo tres niveles de log, ya que consideramos que es suficiente para conseguir separar las distintas trazas. A continuación, enumeramos los niveles empleados y el destino de las trazas:

- **Debug:** Solo las mostraremos en la salida estándar, que en nuestro caso es la consola de ejecución.

- **Info:** Se mostrarán tanto en la salida estándar, como en un fichero de log general. Estas trazas, ayudarán a seguir el hilo de ejecución, y poder conocer la navegación que realizan los usuarios.

- **Error:** Al ser el más importante, se mostrarán en tres sitios. En la salida estándar, en el fichero de log general, y en el fichero de log de errores.

Con este método de organización de trazas, podemos acudir directamente al fichero de errores para ver exclusivamente los errores, y eliminar trazas innecesarias para el seguimiento del mismo, lo que facilita en gran medida las tareas de mantenimiento de la aplicación.

5 SPRINT 1-4

En este capítulo, se recogerán las historias de usuario realizadas en cada uno de los sprints que componen el desarrollo de nuestra página web.

Antes de pasar a ver las historias de usuario implementadas en cada uno de los sprints, consideramos importante explicar el criterio que se ha seguido para establecer la prioridad de cada una de ellas.

Es muy habitual, que tengamos nuestra tarea del día planificada y que venga otra persona a pedirnos realizar una nueva tarea "urgente".

Debemos tener en mente que una tarea que me puede parecer importante, para un compañero o para el PO puede no serlo, de ahí que haya que establecer unos criterios para ser lo más objetivos posibles.

La prioridad que le demos a una tarea no es fija para todo el proyecto, puesto que se debe poner en contexto. Por ejemplo, puede que una tarea planificada para el primer sprint tenga una prioridad baja respecto al resto, sin embargo, si esa misma tarea la planificamos en el segundo o tercer sprint, es muy probable que la prioridad pase a ser bastante alta.

Por ello, a la hora planificar las distintas historias de usuario en los diferentes sprints, es importante detectar posibles dependencias que impidan el desarrollo de una tarea hasta que no finalice otra, lo que puede derivar a un desarrollo descontrolado de la segunda tarea por falta de tiempo.

Cuadrantes de Stephen Covey

Para evitar esta problemática, hemos seguido las directrices marcadas en **los cuadrantes de Stephen Covey**, mundialmente conocido en temas de gestión del tiempo (22).

Ilustración 14. Cuadrantes de Stephen Covey

El autor citado en el párrafo anterior, defiende que "la gente altamente efectiva se enfoca en trabajar sobre el cuadrante 2". Esta afirmación, defiende que lo mejor es realizar aquellas tareas que son importantes, y no urgentes, para los objetivos perseguidos.

De esta manera, evitamos precisamente que una tarea llegue a ser urgente, y como comentábamos anteriormente, provoque estados de crisis y pérdida del control sobre las tareas.

5.1 SPRINT 1

Número: 1	**Nombre: Crear cuenta usuario**
Prioridad: 90	
Descripción: Como usuario, quiero poder registrarme para acceder a la parte privada de la página web.	
Criterio de aceptación: Deben introducirse todos los datos en el formulario de registro con datos válidos, respetando el formato de los campos, y además, debe cumplir los siguientes puntos: - Nombre de usuario único - Email único - Teléfono único	

Tabla 9. HU-1. Crear cuenta usuario

Número: 2	**Nombre: Consultar perfil usuario**
Prioridad: 80	
Descripción: Como usuario, quiero poder consultar los datos de mi perfil para comprobar que son correctos.	
Criterio de aceptación: - En la cabecera, se mostrará el botón "Perfil de 'XXXX'", donde XXXX es el usuario. - Al pulsar ese botón, deberán mostrarse los datos que el usuario introdujo en el formulario de registro.	

Tabla 10. HU-2. Consultar perfil usuario

Número: 3	**Nombre: Modificar perfil usuario**
Prioridad: 80	
Descripción: Como usuario, quiero poder modificar los datos de mi perfil para mantenerlos actualizado.	
Criterio de aceptación: - En la pantalla de consulta del perfil, habrá un botón para acceder a la pantalla de modificación de datos. - En dicha pantalla, se podrán modificarán todos los datos salvo el nombre de usuario. - No se podrá cambiar ni el teléfono ni el email si el nuevo dato ya se encuentra registrado. - Debe introducirse la contraseña como medida de seguridad.	

Tabla 11. HU-3. Modificar perfil usuario

Número: 4	**Nombre: Eliminar cuenta usuario**
Prioridad: 50	
Descripción: Como usuario, quiero poder eliminar mi cuenta para no tener acceso a la página y que no queden guardados mis datos.	
Criterio de aceptación: - Se podrá eliminar tanto en la pantalla de perfil de usuario, como en la pantalla de modificación de datos. - Debe introducirse la contraseña como medida de seguridad - Tras completar el proceso, el usuario no podrá iniciar sesión.	

Tabla 12. HU-4. Eliminar cuenta usuario

Número: 5	**Nombre: Confirmación registro cuenta usuario**
Prioridad: 85	
Descripción: Como PO, quiero que el cliente confirme el registro para asegurarnos de la autoría del email proporcionado.	
Criterio de aceptación: Tras completar el formulario de registro, se enviará un email con un enlace para activar la cuenta. Después, podrá acceder a la parte privada de la web tras iniciar sesión.	

Tabla 13. HU-5. Confirmación registro cuenta usuario

Número: 6	**Nombre: Recuperar contraseña usuario**
Prioridad: 70	
Descripción: Como usuario, quiero poder recuperar o reestablecer mi contraseña para poder seguir accediendo a la página en caso de olvido de la misma.	
Criterio de aceptación: - Tras introducir la dirección de correo que el usuario tiene guardada, se enviará un email con una contraseña de un solo uso. - Tras acceder con esa contraseña, no se podrá volver a acceder con ella.	

Tabla 14. HU-6. Recuperar contraseña usuario

Número: 7	**Nombre: Crear cuenta administrador**
Prioridad: 70	
Descripción: Como usuario, quiero poder registrarme para acceder a la parte privada de la página web para administradores.	
Criterio de aceptación: El registro de un nuevo administrador, lo tendrá que tramitar un administrador vigente.	

Tabla 15. HU-7. Crear cuenta administrador

Número: 8	**Nombre: Consultar perfil administrador**
Prioridad: 60	
Descripción: Como usuario, quiero poder consultar los datos de mi perfil para comprobar que son correctos.	
Criterio de aceptación: Tras pulsar el botón "Perfil", deberán mostrarse los datos que se introdujeron en el formulario de registro.	

Tabla 16. HU-8. Consultar perfil administrador

Número: 9	**Nombre: Modificar perfil administrador**
Prioridad: 60	
Descripción: Como usuario, quiero poder modificar los datos de mi perfil para mantenerlos actualizado.	
Criterio de aceptación: - En la pantalla de consulta del perfil, habrá un botón para acceder a la pantalla de modificación de datos. - En dicha pantalla, se modificarán todos los datos salvo el nombre de usuario. - No se podrá cambiar ni el teléfono ni el email si el nuevo dato ya se encuentra registrado.	

Tabla 17. HU-9. Modificar perfil administrador

Número: 10	Nombre: Eliminar cuenta administrador
Prioridad: 30	
Descripción: Como usuario, quiero poder eliminar mi cuenta para no tener acceso a la página y que no queden guardados mis datos.	
Criterio de aceptación: - Se podrá eliminar tanto en la pantalla de perfil de administrador, como en la pantalla de modificación de datos. - Debe introducirse la contraseña como medida de seguridad - Tras completar el proceso, el usuario no podrá iniciar sesión.	

Tabla 18. HU-10. Eliminar cuenta administrador

Número: 11	Nombre: Confirmación registro cuenta administrador
Prioridad: 85	
Descripción: Como PO, quiero que el administrador confirme el registro para asegurarnos de la autoría del email proporcionado.	
Criterio de aceptación: - Tras completar el formulario de registro, se enviará un email con un enlace para activar la cuenta. Después, podrá acceder a la parte privada de la web tras iniciar sesión.	

Tabla 19. HU-11. Confirmación registro cuenta administrador

Número: 12	Nombre: Recuperar contraseña administrador
Prioridad: 80	
Descripción: Como administrador, quiero poder recuperar o reestablecer mi contraseña para poder seguir accediendo a la página en caso de olvido de la misma.	
Criterio de aceptación: - Tras introducir la dirección de correo que el usuario tiene guardada, se enviará un email con una contraseña de un solo uso. - Tras acceder con esa contraseña, no se podrá volver a acceder con ella.	

Tabla 20. HU-12. Recuperar contraseña administrador

5.2 SPRINT 2

Número: 13	**Nombre: Crear oferta**
Prioridad: 80	
Descripción: Como usuario, quiero poder crear una oferta para que otro usuario pueda verla.	
Criterio de aceptación: Deben introducirse todos los datos en el formulario de registro con datos válidos, respetando el formato de los campos.	

Tabla 21. HU-13. Crear oferta

Número: 14	**Nombre: Ver datos oferta**
Prioridad: 85	
Descripción: Como usuario, quiero poder ver una oferta para consultar los detalles de la oferta.	
Criterio de aceptación: Junto a los datos básicos de la oferta, se pulsará sobre el botón "Ver oferta", y se mostrarán todos los datos que se introdujeron en el formulario al crear la misma.	

Tabla 22. HU-14. Ver datos oferta

Número: 15	**Nombre: Modificar oferta**
Prioridad: 50	
Descripción: Como usuario, quiero poder modificar una oferta para poder actualizar los datos o corregir erratas.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos de la oferta, se pulsará el botón "Modificar oferta", y aparecerá el formulario de modificación de datos. - Deberá introducirse la contraseña como medida de seguridad. - Tras completar el proceso, al volver a consultar los datos, deberá verse el cambio realizado.	

Tabla 23. HU-15. Modificar oferta

Número: 16	**Nombre: Eliminar oferta**
Prioridad: 70	
Descripción: Como usuario, quiero poder eliminar una oferta para que no aparezca en la página.	
Criterio de aceptación: En la pantalla en la que se muestran los datos de la oferta, deberá introducirse la contraseña y pulsar el botón "Eliminar oferta".	

Tabla 24. HU-16. Eliminar oferta

Número: 17	**Nombre: Eliminar ofertas residuales**
Prioridad: 80	
Descripción: Como PO, quiero que se eliminen todas las ofertas de un usuario que elimina su cuenta para evitar mostrar ofertas inválidas.	
Criterio de aceptación: No debe existir ninguna oferta cuyo usuario ha sido eliminado de la base de datos.	

Tabla 25. HU-17. Eliminar ofertas residuales

Número: 18	**Nombre: Ver datos oferta como administrador**
Prioridad: 85	
Descripción: Como administrador, quiero poder ver una oferta para consultar los detalles de la oferta.	
Criterio de aceptación: Al pulsar sobre el botón "Ver oferta", se mostrarán todos los datos registrados en esa oferta.	

Tabla 26. HU-18. Ver datos oferta como administrador

Número: 19	**Nombre: Modificar oferta como administrador**
Prioridad: 50	
Descripción: Como administrador, quiero poder modificar una oferta para poder corregir pequeñas erratas.	
Criterio de aceptación: En la pantalla en la que se muestran los datos de la oferta, se pulsará el botón "Modificar oferta", y aparecerá el formulario de modificación de datos.	

Tabla 27. HU-19. Modificar oferta como administrador

Número: 20	Nombre: Listar ofertas pendientes
Prioridad: 90	
Descripción: Como administrador, quiero poder ver todas las ofertas pendientes para poder activarlas o rechazarlas.	
Criterio de aceptación: En la sección de "Gestionar ofertas": - Se pulsará el botón "Ofertas pendientes". - Se mostrarán exclusivamente las ofertas cuyo estado sea "Sin activar".	

Tabla 28. HU-20. Listar ofertas pendientes

Número: 21	Nombre: Listar ofertas rechazadas
Prioridad:40	
Descripción: Como administrador, quiero poder ver todas las ofertas rechazadas para poder tener un control sobre ellas.	
Criterio de aceptación: En la sección de "Gestionar ofertas": - Se pulsará el botón "Ofertas rechazadas". - Se mostrarán exclusivamente las ofertas cuyo estado sea "Rechazada".	

Tabla 29. HU-21. Listar ofertas rechazadas

Número: 22	Nombre: Activar oferta
Prioridad: 90	
Descripción: Como administrador, quiero poder activar una oferta para que todos los usuarios tengan acceso a ella desde la parte pública.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos de la oferta, se pulsará el botón "Activar oferta". - Cuando se vuelvan a consultar los datos de la oferta, aparecerá con el estado "Activa" y la fecha de fin.	

Tabla 30. HU-22. Activar oferta

Número: 23	**Nombre: Rechazar oferta**
Prioridad: 70	
Descripción: Como administrador, quiero poder rechazar una oferta para que evitar publicar ofertas inadecuadas.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos de la oferta, aparecerá un combo para seleccionar el motivo del rechazo y se pulsará el botón "Rechazar oferta". - Cuando se vuelvan a consultar los datos de la oferta, aparecerá con el estado "Rechazada".	

Tabla 31. HU-23. Rechazar oferta

Número: 24	**Nombre: Expirar ofertas finalizadas**
Prioridad: 90	
Descripción: Como PO, quiero que las ofertas finalizadas se marquen como expiradas para que los usuarios no puedan tener acceso a ellas.	
Criterio de aceptación: Las ofertas cuya fecha de fin sea anterior a la fecha actual, deberán cambiar automáticamente su estado a "Expirada".	

Tabla 32. HU-24. Expirar ofertas finalizadas

5.3 SPRINT 3

Número: 25	**Nombre: Listar ofertas activas**
Prioridad:90	
Descripción: Como usuario, quiero poder ver el listado de las ofertas activas para saber qué ofertas están disponibles.	
Criterio de aceptación: En la página principal, tanto en la parte pública como en la privada, solo se mostrarán las ofertas cuyo estado sea "Activa".	

Tabla 33. HU-25. Listar ofertas activas

Número: 26	**Nombre: Listar ofertas propias**
Prioridad:80	
Descripción: Como usuario, quiero poder ver las ofertas que he creado para tener un acceso más rápido a su gestión.	
Criterio de aceptación: - En la cabecera, aparecerá un botón llamado "Mis ofertas". - Al pulsarlo, se mostrarán exclusivamente las ofertas creadas por ese usuario.	

Tabla 34. HU-26. Listar ofertas propias

Número: 27	**Nombre: Contactar con usuario**
Prioridad: 80	
Descripción: Como usuario, quiero poder contactar con el creador de una oferta para poder cerrar el trato.	
Criterio de aceptación: - En la pantalla del detalle de la oferta, se mostrará el botón "Contactar usuario". - Al pulsarlo, se enviará un email al creador de la oferta con los datos de contacto del usuario interesado.	

Tabla 35. HU-27. Contactar con usuario

Número: 28	**Nombre: Buscar por categoría**
Prioridad:75	
Descripción: Como usuario, quiero poder seleccionar una categoría para ver solo las ofertas de esa categoría.	
Criterio de aceptación: - En el menú, habrá una sección llamada "Ver ofertas" donde se desplegarán las diferentes categorías. - Al seleccionar una categoría, solo deberán mostrarse ofertas pertenecientes a la categoría elegida.	

Tabla 36. HU-28. Buscar por categoría

Número: 29	**Nombre: Buscar usuario**
Prioridad: 60	
Descripción: Como administrador, quiero poder buscar un usuario para poder realizar gestiones sobre él.	
Criterio de aceptación: - En la sección "Gestionar usuarios", habrá un campo para introducir el usuario que se quiere buscar. - Si no existe, se muestra un mensaje informativo.	

Tabla 37. HU-29. Buscar usuario

Número: 30	**Nombre: Buscar administrador**
Prioridad: 50	
Descripción: Como administrador, quiero poder buscar un administrador para poder realizar gestiones sobre él.	
Criterio de aceptación: - En la sección "Gestionar usuarios", habrá un campo para introducir el administrador que se quiere buscar. - Si no existe, se muestra un mensaje informativo.	

Tabla 38. HU-30. Buscar administrador

Número: 31	**Nombre: Ver datos de usuario**
Prioridad: 50	
Descripción: Como administrador, quiero poder consultar los datos de un usuario para poder realizar gestiones sobre él.	
Criterio de aceptación: Tras buscar un usuario existente en la sección "Gestionar usuarios", se mostrarán los datos del mismo.	

Tabla 39. HU-31. Ver datos usuario

Número: 32	**Nombre: Ver datos de administrador**
Prioridad: 50	
Descripción: Como administrador, quiero poder consultar los datos de un administrador para poder realizar gestiones sobre él.	
Criterio de aceptación: Tras buscar un administrador existente en la sección "Gestionar usuarios", se mostrarán los datos del mismo.	

Tabla 40. HU-32. Ver datos de administrador

Número: 33	**Nombre: Bloquear usuario**
Prioridad: 70	
Descripción: Como administrador, quiero poder bloquear un usuario para prohibirle temporalmente el acceso a la aplicación.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos del usuario, se mostrará un combo donde elegir el periodo de bloqueo. - Tras bloquear al usuario, este no podrá acceder a la parte privada de la página web mientras la fecha de bloqueo sea posterior a la fecha actual.	

Tabla 41. HU-33. Bloquear usuario

Número: 34	Nombre: Desbloquear usuario
Prioridad: 70	
Descripción: Como administrador, quiero que se desbloquee un usuario bloqueado para que pueda volver a acceder a la parte privada de la web.	
Criterio de aceptación: Cuando un usuario bloqueado intente iniciar sesión y la fecha de bloqueo sea anterior a la fecha actual, se desbloqueará su cuenta y podrá acceder a la parte privada de la página web.	

Tabla 42. HU-34. Desbloquear usuario

Número: 35	Nombre: Eliminar usuario
Prioridad: 70	
Descripción: Como administrador, quiero poder eliminar un usuario para prohibirle negarle el acceso a la aplicación indefinidamente.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos del usuario, aparecerá el botón "Eliminar usuario". - No quedarán registrados los datos de ese usuario en la base de datos.	

Tabla 43. HU-35. Eliminar usuario

Número: 36	Nombre: Eliminar administrador
Prioridad: 80	
Descripción: Como administrador, quiero poder eliminar un administrador para negarle el acceso a la aplicación indefinidamente.	
Criterio de aceptación: - En la pantalla en la que se muestran los datos del administrador, aparecerá el botón "Eliminar administrador". - No quedarán registrados los datos de ese administrador en la base de datos.	

Tabla 44. HU-36. Eliminar administrador

Número: 37	Nombre: Ver resumen de ofertas de un usuario
Prioridad: 40	
Descripción: Como administrador, quiero poder ver un resumen numérico de las ofertas de un usuario para ver la actividad del mismo.	
Criterio de aceptación: En la pantalla en la que se muestran los datos del usuario, se mostrará el número de ofertas activadas, pendientes, rechazadas y expiradas.	

Tabla 45. HU-37. Ver resumen de ofertas de un usuario

Número: 38	Nombre: Ver resumen de usuarios
Prioridad: 50	
Descripción: Como administrador, quiero poder ver un resumen numérico de los usuarios y administradores registrados en la aplicación para poder saber el número de usuarios de la misma.	
Criterio de aceptación: En la pantalla de inicio de la aplicación para administradores, se mostrará el número de usuarios y administradores registrados, y cuántos de ellos han confirmado el registro.	

Tabla 46. HU-38. Ver resumen de usuarios

Número: 39	Nombre: Ver resumen de ofertas
Prioridad: 40	
Descripción: Como administrador, quiero poder ver un resumen numérico de las ofertas existentes para saber en qué estado están (activas, pendientes, etc.).	
Criterio de aceptación: En la pantalla de inicio de la aplicación para administradores, se mostrará el número total de ofertas activas, pendientes, rechazadas y expiradas.	

Tabla 47. HU-39. Ver resumen de ofertas

5.4 SPRINT 4

Número: 40	Nombre: Preguntas frecuentes
Prioridad: 40	
Descripción: Como usuario, quiero poder acudir a preguntas frecuentes para resolver dudas comunes.	
Criterio de aceptación: Tras acceder a la sección "Preguntas frecuentes" del menú, se mostrará: - En la parte superior, los enunciados de las preguntas frecuentes. - Enlazadas al clicar sobre el enunciado, se encontrarán las respuestas a dichas preguntas.	

Tabla 48. HU-40. Preguntas frecuentes

Número: 41	Nombre: Página de contacto
Prioridad: 50	
Descripción: Como usuario, quiero poder enviar consultas para resolver dudas que no se resuelvan en las preguntas frecuentes.	
Criterio de aceptación: Tras acceder a la sección "Contacta con nosotros", se mostrarán los datos de contacto a los que se pueden dirigir los usuarios para enviar sus dudas personales.	

Tabla 49. HU-41. Página de contacto

Número: 42	Nombre: TOP valoración
Prioridad: 60	
Descripción: Como usuario, quiero poder ver el listado de las ofertas mejor valoradas para facilitarme la búsqueda de buenas ofertas.	
Criterio de aceptación: Al acceder a la sección "TOP 10 > Mejor valoradas", se mostrarán las 10 ofertas con mayor valoración acumulada.	

Tabla 50. HU-42. TOP valoración

Número: 43	**Nombre: TOP visitas**
Prioridad: 60	
Descripción: Como usuario, quiero poder ver el listado de las ofertas con más visitas para saber qué es lo que más busca la gente.	
Criterio de aceptación: Al acceder a la sección "TOP 10 > Más visitadas", se mostrarán las 10 ofertas con mayor número de visitas.	

Tabla 51. HU-43. TOP visitas

Número: 44	**Nombre: Puntuar ofertas**
Prioridad: 65	
Descripción: Como usuario, quiero poder puntuar una oferta para ayudar a otros usuarios en su búsqueda de buenas ofertas.	
Criterio de aceptación: En la pantalla en la que se muestran los detalles de la oferta: - Se mostrará un combo para seleccionar la puntuación que le queremos dar a la oferta, siempre que no lo hayamos hecho antes. - Si ya hemos puntuado la oferta, se mostrará un mensaje informativo en lugar del combo.	

Tabla 52. HU-44. Puntuar ofertas

Número: 45	**Nombre: Ordenación de ofertas por fecha**
Prioridad: 50	
Descripción: Como usuario, quiero poder ver las ofertas ordenadas por fecha de finalización para ver rápidamente cuáles son aquellas que expiran antes.	
Criterio de aceptación: - En la página principal, tanto en la parte pública como en la privada, se mostrarán las ofertas ordenadas por fecha de finalización. - Deberá mostrarse en primer lugar aquella oferta a la que le quede menos tiempo de actividad.	

Tabla 53. HU-45. Ordenación de ofertas por fecha

Número: 46	Nombre: Buscar ofertas por identificador
Prioridad: 50	
Descripción: Como administrador, quiero poder buscar una oferta introduciendo su identificador para facilitarme el proceso de búsqueda.	
Criterio de aceptación: - En la sección "Gestionar ofertas", se podrá introducir el identificador de una oferta. - Si existe, se mostrarán los datos de la misma. - Si no existe, se mostrará un mensaje informativo.	

Tabla 54. HU-46. Buscar ofertas por identificador

Número: 47	Nombre: Buscar ofertas por usuario
Prioridad: 60	
Descripción: Como administrador, quiero poder ver todas las ofertas de un usuario introduciendo el nombre de usuario para facilitarme el proceso de búsqueda.	
Criterio de aceptación: - En la sección "Gestionar ofertas", se podrá introducir el nombre de un usuario. - Si existe y tiene ofertas, se mostrará el listado de ofertas creadas por dicho usuario. - En caso contrario, se mostrará un mensaje informativo.	

Tabla 55. HU-47. Buscar ofertas por usuario

Número: 48	Nombre: Periodo activación manual
Prioridad: 70	
Descripción: Como PO, quiero que el periodo de actividad de una oferta sea manual para poder establecer periodos distintos en base al tipo de producto.	
Criterio de aceptación: En la misma pantalla en la que se activan las ofertas, se deberá introducir el número de días/meses que estará activa la oferta.	

Tabla 56. HU-48. Periodo activación manual

Número: 49	Nombre: Información sobre nosotros
Prioridad: 50	
Descripción: Como PO, quiero poder ver información sobre la empresa para que el cliente pueda saber quiénes somos.	
Criterio de aceptación: En la sección "Sobre nosotros" del menú, deberá aparecer información sobre la empresa.	

Tabla 57. HU-49. Información sobre nosotros

Número: 50	Nombre: Publicidad
Prioridad: 30	
Descripción: Como PO, quiero mostrar publicidad externa para tener ingresos.	
Criterio de aceptación: Habrá una sección en la parte privada de la aplicación, en la que aparecerá el formulario para registrar los datos el anuncio.	

Tabla 58. HU-50. Publicidad

Número: 51	Nombre: Mostrar datos publicidad
Prioridad: 30	
Descripción: Como administrador, quiero poder ver los datos de un anuncio para ver los detalles.	
Criterio de aceptación: Habrá una sección en la parte privada de la aplicación, en la que se mostrarán las ofertas registradas.	

Tabla 59. HU-51. Mostrar datos publicidad

Número: 52	Nombre: Activar publicidad
Prioridad: 30	
Descripción: Como administrador, quiero poder activar la publicidad a mostrar para cumplir con los acuerdos con el anunciante.	
Criterio de aceptación: En la pantalla en la que se muestran los datos del anuncio, se podrá activar el anuncio.	

Tabla 60. HU-52. Activar publicidad

Número: 53	Nombre: Expirar publicidad
Prioridad: 30	
Descripción: Como PO, quiero que las ofertas expiren al finalizar el periodo para no mostrarla fuera de plazo.	
Criterio de aceptación: Si la fecha de fin es anterior a la fecha actual, deberá expirar la oferta para no mostrarla más.	

Tabla 61. HU-53. Expirar publicidad

6 CONCLUSIONES Y TRABAJOS FUTUROS

En este capítulo, hablaremos sobre las conclusiones obtenidas acerca de los objetivos perseguidos durante la realización del trabajo de fin de grado, que como veremos, han sido bastante positivas. Además, comentaremos posibles trabajos futuros o mejoras, partiendo de la aplicación generada en el mismo.

6.1 Conclusiones

El proyecto aquí presentado, constituye una base sobre la que seguir construyendo y adquiriendo conocimientos en este sector tan cambiante.

El principal objetivo del trabajo de fin de grado que nos ocupa, era conseguir desarrollar un web site que permitiera a los usuarios publicar anuncios que quisieran vender, al tiempo que pusiera en contacto a las personas interesadas con el propietario de la oferta.

Además de cumplir esas funcionalidades básicas, se ha dotado a la aplicación de otras características para ayudar al usuario a dar salida a las ofertas, como por ejemplo, la inclusión de imágenes de los productos, sistemas de puntuación de ofertas, etc.

Con el objetivo de facilitar la administración del web site, se ha desarrollado de manera paralela una aplicación, para que sea accedida exclusivamente por los administradores desde una intranet. Desde aquí, podrán gestionar tanto a los usuarios como a las ofertas. La utilización de las tecnologías aplicadas al proyecto, contribuyen a establecer una base estable sobre la que seguir mejorando la aplicación, adaptándose a las futuras novedades tecnológicas.

Seguidamente, nos marcamos como objetivo conocer las características de las metodologías de desarrollo tradicional y las de las metodologías ágiles, para poder elegir con criterio aquella que nos proporcionara adaptabilidad para poder afrontar cambios inesperados.

Este objetivo, me ha permitido conocer el movimiento ágil, complementando la visión adquirida en los años de estudio en la universidad sobre las metodologías de desarrollo.

Finalmente, nos propusimos como objetivo utilizar herramientas y mecanismos que añadiesen valor a la aplicación generada. Durante el desarrollo del trabajo de fin de grado, hemos aprendido a utilizar algunas herramientas muy interesantes, como por ejemplo, realizar pruebas unitarias y de regresión, de manera automática con test de JUnit, o utilizar sistemas de

logging de trazas, para facilitar las tareas de mantenimiento de la aplicación. Además, gracias al estudio realizado sobre los test unitarios, hemos aprendido cómo aplicar la técnica de desarrollo basado en las pruebas (TDD),

Estos mecanismos que acabamos de mencionar, a pesar de no añadir funcionalidad a la propia aplicación, y pasar desapercibidas para el usuario final que accederá a nuestro web site, incrementa en gran medida la calidad final del producto.

Por último, he de destacar que tras la realización del trabajo de fin de grado, me siento muy satisfecho del trabajo realizado, y de los conocimientos adquiridos en él, tanto técnicos como conceptuales, y que seguro me servirán a lo largo de mi carrera profesional.

6.2 Trabajos futuros

Tras la realización del trabajo de fin de grado, se plantean varios aspectos sobre los que seguir investigando en trabajos futuros:

- Mejorar la **accesibilidad** de la página web para proporcionar una experiencia adecuada a gente con problemas visuales.

- Siguiendo la línea del punto anterior, se podría mejorar el diseño de la aplicación para que sea adaptativo al dispositivo empleado, siguiendo las pautas del **responsive design.**

- Otra posibilidad interesante, es el uso de otros frameworks de desarrollo como pueden ser Struts o Spring Webflow, que junto con el concepto SPI tratado en este documento, puede suponer una estupenda mejora para nuestra aplicación.

- Siguiendo con las mejoras visuales, se podría trabajar con librerías que nos aporten efectos con los que mostrar una aplicación mucho más dinámica, como por ejemplo AJAX o librerías añadidas de JQuery.

- Otra línea de investigación que está cogiendo fuerza en los últimos años, es el uso de NodeJS en la parte backend para hacer de servidor de una aplicación web, basado en lenguaje Javascript, en lugar de los servidores tradicionales como el nuestro basado en Java.

- Respecto a la base de datos, se están usando cada vez más bases de datos NoSQL, que en vez de almacenar los datos tablas, como estamos acostumbrados, lo hacen en documentos JSON, lo que permite una adaptación más rápida para aplicaciones que trabajan con ese formato.

7 GESTIÓN DEL PROYECTO

En este capítulo, se tratarán los temas relativos a la gestión del trabajo de fin de grado. Se hablará de la planificación de las fases a realizar y de se mostrará un presupuesto estimado.

7.1 Planificación

En este apartado, se mostrará una tabla en la que aparecen las fechas de las fases a realizar en el desarrollo del proyecto.

Fase	Fecha de inicio	Fecha final	Días
Planificación	30/09/2013	04/10/2013	5 días
Estudio previo	07/10/2013	20/10/2013	14 días
Sprint 0	21/10/2013	03/11/2013	14 días
Sprint 1	04/11/2013	24/11/2013	21 días
Sprint 2	25/11/2013	15/12/2013	21 días
Sprint 3	16/12/2013	05/01/2014	21 días
Sprint 4	08/01/2014	28/01/2014	21 días
Memoria y documentación	01/02/2014	23/02/2014	23 días
Total	**01/10/2013**	**23/02/2014**	**140 días**

Tabla 62. Planificación del proyecto

En base a los datos mostrados en la tabla, pasamos a realizar el cálculo del tiempo empleado en el conjunto del trabajo de fin de grado.

Dividiendo el total de días en semanas, obtenemos un total de 20 semanas. Teniendo en cuenta que por motivos de trabajo se ha dedicado una media de 15 horas semanales, nos sale que se han dedicado 300 horas a la realización del proyecto, tiempo que se ajusta a los 12 créditos correspondientes al trabajo de fin de grado.

7.2 Presupuesto

En este apartado, se realizará el cálculo del presupuesto en base al trabajo realizado en este proyecto. Para calcularlo, hay que tener en cuenta los siguientes factores:

- Horas de trabajo
- Elementos hardware
- Licencias software

En cuanto a las licencias software, no aplica en nuestro trabajo de fin de grado, puesto que todo el software empleado es de código libre y por tanto tiene coste 0.

Del mismo modo, puesto que se trata de una aplicación docente, que por ahora no va a tener tráfico real, los elementos hardware no van a incrementar el coste del mismo. En esta ocasión, tanto el servidor de aplicaciones como la base de datos, estarán alojados en un ordenador personal y por tanto, el coste de este apartado corresponderá a la valoración de este equipo.

El equipo utilizado para el desarrollo del trabajo de fin de grado, es un ordenador portátil modelo HP Pavilion dv6 Notebook PC, con procesador Intel Core i7, 8 GB de RAM y sistema operativo Windows 7 Home Premium, valorado en 700 €.

En caso de sacar a la luz la aplicación, habría que tener en cuenta el tráfico y los usuarios concurrentes para calcular la capacidad del servidor (o servidores junto con un balanceador), y los costes añadidos como puede ser el precio anual del dominio de la web.

Una alternativa a la compra de estos componentes, que resultaría en alto coste inicial, sería el **alojamiento web.** Dependiendo de la modalidad escogida (servidor dedicado o compartido), podemos obtener un precio anual bastante económico y seguir cubriendo las necesidades de nuestra aplicación.

Asumiendo la modalidad de **alojamiento compartido**, en el que un mismo servidor aloja la aplicación de varios clientes, supondría una cuota aproximada de 100 € anuales con dominio incluido.

Por último, lo que representará la mayor parte del presupuesto en nuestro caso, son las horas de trabajo realizadas en el mismo. En este apartado hay que tener en cuenta tanto las horas indicadas en la tabla anterior, que corresponden con las horas de desarrollo y arquitectura, como las horas dedicadas por el PO.

En nuestro caso, el PO ha intervenido tres horas por cada uno de los sprints, y cuatro en el sprint 0, nos sale un total de 16 horas. Asumiendo que el precio por hora del PO es de 30 €, tenemos un subtotal de 480 €.

Por otro lado, se han empleado 300 horas para el resto de tareas. Estableciendo un precio de 21 €/hora, esta parte del presupuesto supondría un subtotal de 6.300 €.

Por tanto, el presupuesto total del trabajo de fin de grado es de **7.480 €**, más 100 €/año si elegimos el alojamiento web compartido.

GLOSARIO

En este apartado, se recogerán aquellos términos técnicos (o poco conocidos) que aparecen a lo largo de este documento para poder seguir el hilo argumental de este documento. Para facilitar su localización, se listarán en orden alfabético.

- **AJAX:** Se trata de una herramienta de desarrollo web que permite modificar algunas partes de la página sin necesidad de recargarla.

- **Alojamiento web:** Es un servicio que provee a los usuarios un servidor en el que alojar una aplicación web, de manera que se evita tener que comprar equipos propios.

- **Backend:** Parte del lado del servidor de una aplicación web. Es la encargada la parte en cargada de comunicarse con la base de datos y de tratar los datos que recibe desde la parte frontend.

- **Backlog:** Es un documento empleado en las metodologías ágiles, que recoge el conjunto de tareas que ha de realizarse, ordenadas por prioridad.

- **DOM:** Document Object Model. Es una interfaz de programación que proporciona un estándar de objetos para representar documentos HTML, y permitir modificar el contenido de los mismos.

- **Feedback:** Información que se recibe después de revisar las funcionalidades aportadas. De él, se pueden extraer detalles para mejorar de cara a futuras entregas.

- **Framework:** Es un conjunto de herramientas y módulos que ayudan al desarrollo de aplicaciones.

- **Frontend:** Parte del lado del cliente de una aplicación web. Es la encargada de la maquetación de la aplicación, y de realizar la interacción con el usuario.

- **JQuery:** Es una librería de Javascript, que simplifica en gran medida la interacción con el contenido HTML.

- **JSON:** Es un formato ligero para intercambio de datos. Fácil de interpretar tanto por la máquina como por las personas.

- **JSP:** Siglas de Java Server Page. Son ficheros de la parte frontend, en los que se utiliza tanto código HTML como código Java, de que el contenido de la página varíe según ciertos condiciones.

- **NodeJS:** Es un entorno de programación en el lado del servidor, basado en lenguaje Javascript.

- **Pila del producto:** Lista de tareas a realizar, que estarán recogidas en el backlog del producto.

- **Pila del sprint:** Lista de tareas a realizar dentro de un sprint, y que estarán recogidas en el backlog del sprint.

- **Product owner (PO):** Es el responsable del producto en los equipos ágiles. Debe conocer el negocio del cliente, y es el encargado de determinar las prioridades de las tareas recogidas en el backlog.

- **Responsive design:** Es un tipo de desarrollo web, que permite adaptar el contenido que se muestra al usuario al tamaño de la pantalla del dispositivo empleado.

- **Servlet:** Es una clase Java, que sirve para redirigir el flujo de navegación en base a las peticiones realizadas durante la navegación del usuario.

- **Scrum master (SM):** Persona responsable del correcto funcionamiento de scrum en el proyecto. Es el encargado de formar, entrenar, supervisar y facilitar la realización de las prácticas de scrum.

- **Spring:** Es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java.

- **Sprint:** Periodo de tiempo en el que se desarrolla una iteración del producto. Es recomendable que se establezca entre 1 semana y 1 mes, dependiendo de la complejidad del proyecto.

- **Struts:** Es un framework que da soporte para desarrollos de aplicaciones para la plataforma Java EE.

- **TDD:** Test Driven Development. Técnica de desarrollo a partir de las pruebas unitarias.

- **Time to market:** Tiempo que tarda un producto desde que es concebido hasta que llega al mercado.

- **Web site:** Es un conjunto de páginas web interconectadas, comunes a un mismo dominio.

BIBLIOGRAFÍA

1. ANDERSON, Paul. Entienda la web 2.0 y sus principales servicios. En Eduteka: *portal educativo que fomenta el uso de las tecnologías de la información.* [Consulta: 01-02-2014]. Disponible en: http://www.eduteka.org/Web20Intro.php

2. Wikipedia. [Consulta: 01-02-2014]. Disponible en: www.es.wikipedia.org/

3. Duiops. *Mercadotecnia de los sitios Web*. [Consulta: 01-02-2014]. Disponible en: http://www.duiops.net/curso/mercadot.htm

4. ABC. *Así han cambiado nuestros hábitos de navegación.* [Consulta: 02-02.2014]. Disponible en: http://www.abc.es/20101210/medios-redes/habitos-lectura-internet-201012101526.html

5. GUTIÉRREZ, Pedro. Responsive Design: introducción. En Genbetadev: *blog sobre desarrollo y software.* [Consulta: 03-02-2014]. Disponible en: http://www.genbetadev.com/desarrollo-web/responsive-design-introduccion

6. W3C. *Introducción a la Accesibilidad Web.* [Consulta: 03-02-2014]. Disponible en: http://www.w3c.es/Traducciones/es/WAI/intro/accessibility

7. ARRANZ Santamaría, José María. *Manifiesto por una Única Página Web (Single Page Interface).* [Consultado: 21-02-2014]. Disponible en: http://itsnat.sourceforge.net/php/spim/spi_manifesto_es.php

8. CORRAL, Rodrigo. ¿Qué metodología de desarrollo elegir?. En Geeks.ms: *blog experiencias de programación y desarrollo software.* [Consulta: 04-02-2014]. Disponible en: http://geeks.ms/blogs/rcorral/archive/2007/01/15/iquest-que-metodolog-iacute-a-de-desarrollo-elegir.aspx

9. INTECO. *Ingeniería del Software: Metodologías y ciclos de vida.* [en línea] [Consulta: 04-02-2014]. Disponible en: https://www.google.es/url?sa=t&rct=j&q=&esrc=s&source=web&cd=12&cad=rja&ved=0CGMQFjAL&url=https%3A%2F%2Fwww.inteco.es%2Ffile%2FN85W1ZWFHifRgUc_oY8_Xg&ei=QeUIU8z6LYWk0QWCu4E4&usg=AFQjCNFgTCy909i9uzT-_5w2VGCJ7zCf9g&bvm=bv.61725948,d.d2k

10. NOVILLO Guijarro, Ana. El pensamiento lean y la eliminación de desperdicios en la gestión. En Infoautónomos: *blog en el que se ayuda a autónomos gestionar su empresa.* [Consulta: 20-02-2014]. Disponible en: http://www.infoautonomos.com/blog/el-pensamiento-lean-y-la-eliminacion-de-desperdicios-en-la-gestion/

11. Agilemanifesto. *Manifiesto por el Desarrollo Ágil de Software.* [Consultado: 06-02-2014]. Disponible en: http://agilemanifesto.org/iso/es/

12. Proyectosagiles. *Qué es SCRUM.* [Consultado: 06-02-2014]. Disponible en: http://www.proyectosagiles.org/que-es-scrum

13. Softeng. *Proceso y Roles de Scrum.* [Consultado: 07-02-2014]. Disponible en: http://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html

14. QUIJANO, Juan. Historias de usuario, una forma natural de análisis funcional. En Genbetadev: *blog sobre desarrollo y software.* [Consulta: 07-02-2014]. Disponible en: http://www.genbetadev.com/metodologias-de-programacion/historias-de-usuario-una-forma-natural-de-analisis-funcional

15. BEAS, José Manuel. Agile Inception. En JMBEAS: *blog sobre desarrollo de software.* [Consultado: 20-02-2014]. Disponible en: http://jmbeas.es/guias/agile-inception/

16. EjemplosTIW. *Patrón de arquitectura Modelo Vista Controlador (MVC).* [Consulta: 10-02-2014]. Disponible en: http://www.lab.inf.uc3m.es/~a0080802/RAI/mvc.html

17. HASSAN Montero, Yusef. Introducción a la Usabilidad. En No solo usabilidad: *revista sobre personas, diseño y tecnología.* [Consulta: 12-02-2014]. ISSN 1886-8592. Disponible en: http://www.nosolousabilidad.com/articulos/introduccion_usabilidad.htm

18. ScrumManager. *Refactorización del código.* [Consultado: 13-02-2014]. Disponible en: http://www.scrummanager.net/bok/index.php?title=Refactorizaci%C3%B3n_del_c%C3%B3digo

19. BENÍTEZ, Carlos. Test Unitarios. Cuándo usarlos y pistas para conseguir un sistema robusto. En etnassoft: *página web personal sobre programación y testing.* [Consultado: 13-02-2015]. Disponible en: http://www.etnassoft.com/2011/07/27/tests-unitarios-cuando-usarlos-y-pistas-para-conseguir-un-sistema-robusto/

20. DEL BARRIO, Oriol. Introducción a TDD (Test Driven Development). En Lordudun blog: *blog sobre tecnología y desarrollo de software.* [Consulta: 19-02-2014]. Disponible en: http://blog.lordudun.es/2011/04/introduccion-a-tdd-test-driven-development/

21. ACEDO, Jose I. Log4j: Tutorial y configuración rápida. En Apuntes de Programación: *blog de programación.* [Consultado: 13-02-14]. Disponible en: http://programacion.jias.es/2013/03/log4j-tutorial-configuracion-rapida/#more-1418

22. FIGUEROLA, Norberto. Lo importante y lo urgente en la Dirección de Proyectos. En Líder de proyecto: *página sobre administración de proyectos.* [Consultado: 20-02-2014]. Disponible en: http://www.liderdeproyecto.com/articulos/16_lo_urgente.html

ANEXO I

En este anexo, se incluirá una breve descripción y las capturas de pantalla más importantes de algunos de los programas utilizados, para que sirvan de ayuda al lector en caso de que está interesado en utilizarlos.

Glassfish

Se trata de un servidor de aplicaciones gratuito, desarrollado por Sun Microsystems, que permite la ejecución de aplicaciones Java EE. Dispone una consola administrativa, que permite configurarlo de manera sencilla.

Instalación

Para el desarrollo de este trabajo de fin de grado, ya que tenía que utilizar un JDK para JAVA, utilicé un instalador de JDK 6 + Glassfish disponible en la página de descargas de Oracle. En este caso, solo una de las pantallas de instalación es importante destacar. En ella, se pondrá el puerto HTTP en el que el servidor esperará las peticiones, el puerto de acceso a la consola del administrador, y la contraseña, que habrá que introducir tanto para acceder a la consola, como para arrancar o eliminar una aplicación asociada al servidor.

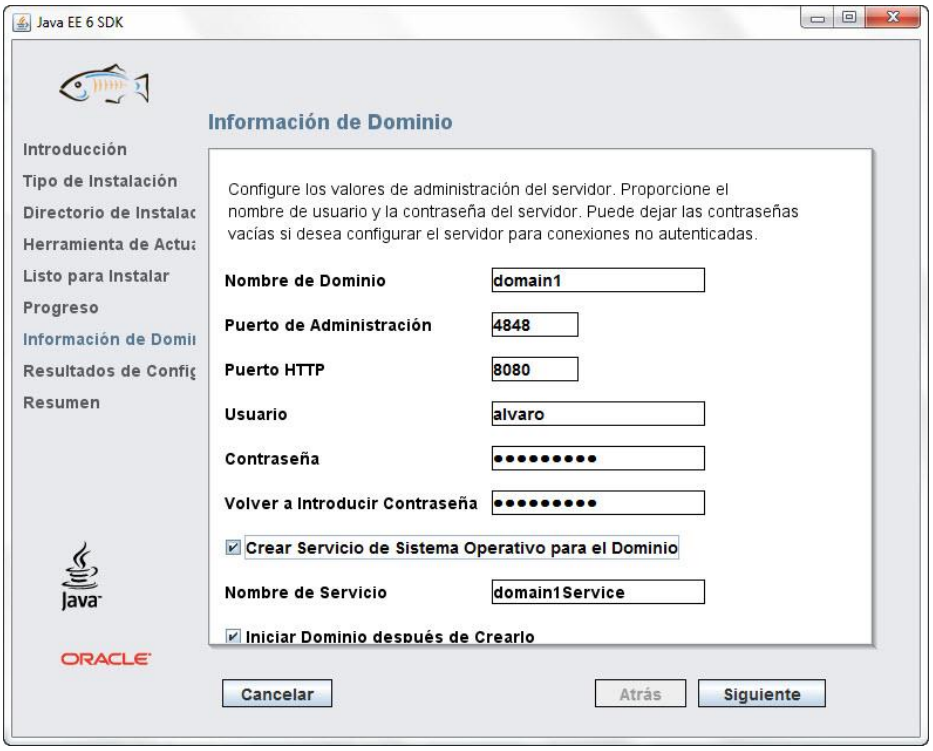

Ilustración 15. Configuración Glassfish

Integración en Eclipse

Una vez instalado, solo queda incluirlo en nuestro entorno de Eclipse para asociarle la aplicación web que estemos desarrollando. Para ello, nos dirigimos a la vista de servidores, hacemos click derecho, y seleccionamos 'New>Server'.

En la ventana que nos aparece, buscamos la carpeta 'Glassfish' y la versión. Además, podremos elegir el nombre con el que aparecerá el servidor en Eclipse, por si tuviéramos más de un servidor y los quisiéramos diferenciar.

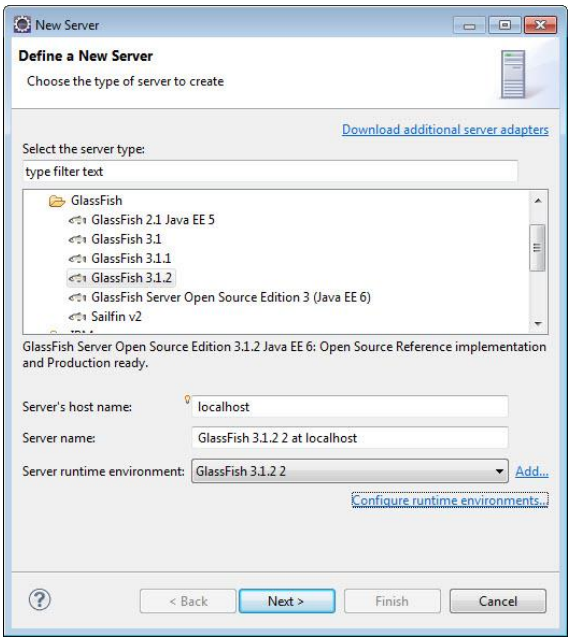

Ilustración 16. Integración de Glassfish en Eclipse

Una vez finalizado el proceso, aparecerá nuestro nuevo servidor en la vista de servidores de Eclipse.

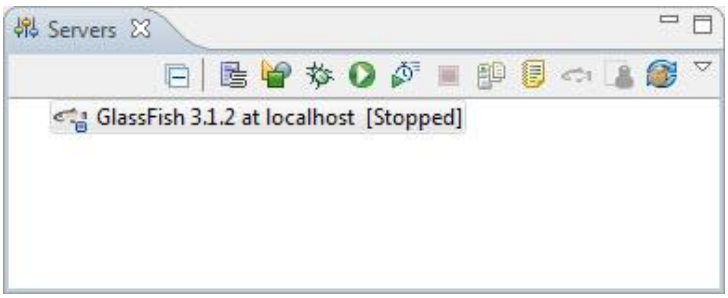

Ilustración 17. Glassfish integrado

MySQL Server

Para poder alojar la base de datos en nuestro propio ordenador, es necesario instalar este programa para que haga de servidor para que reciba las peticiones sobre la misma.

Tras instalarlo, hay que configurar la instancia del servidor, es decir, la configuración del puerto, el número de conexiones, etc. La información que se muestra en las diferentes pantallas de configuración, son bastante explicativas y ayudan realmente al usuario. Aun así, ajunto la configuración que elegí para el desarrollo de este proyecto.

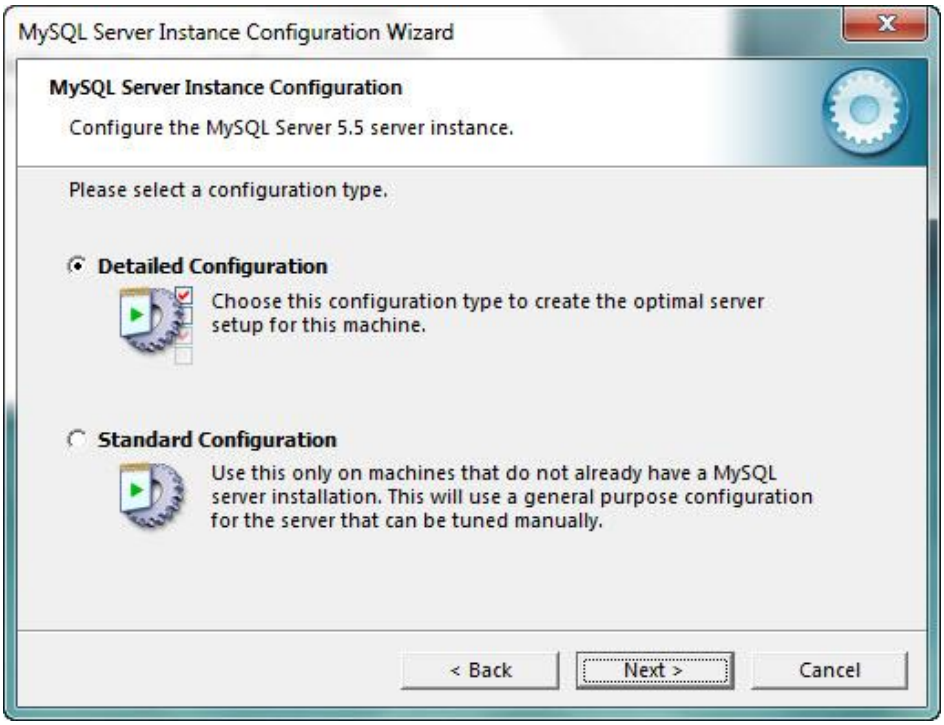

Ilustración 18. MySQL Server. Configuración detallada

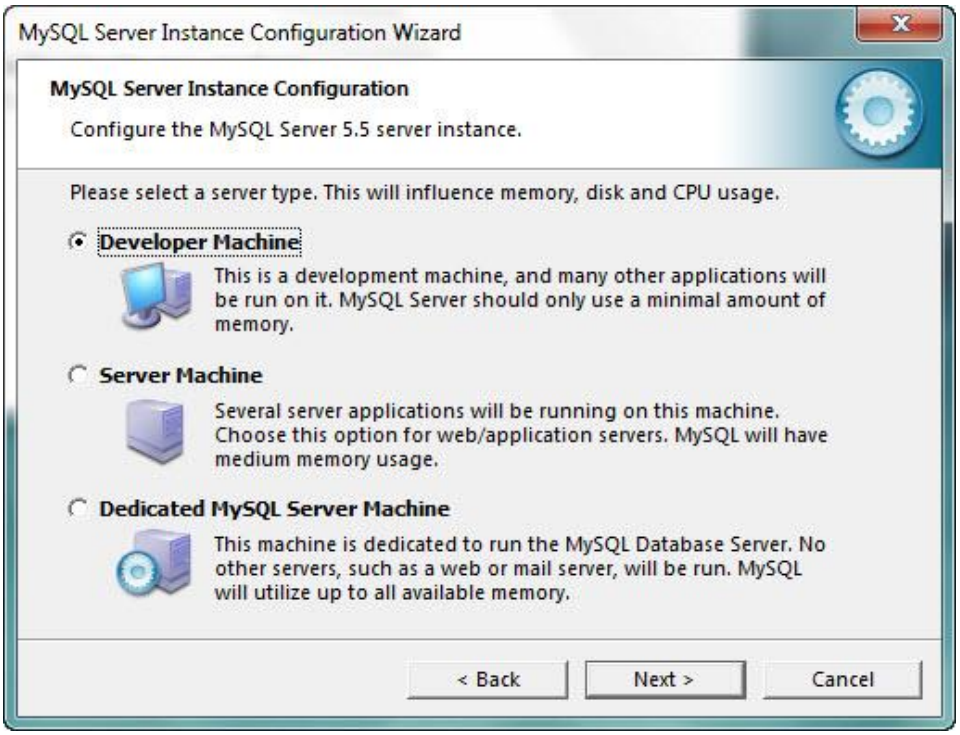

Ilustración 19. MySQL Server. Servidor para desarrollar

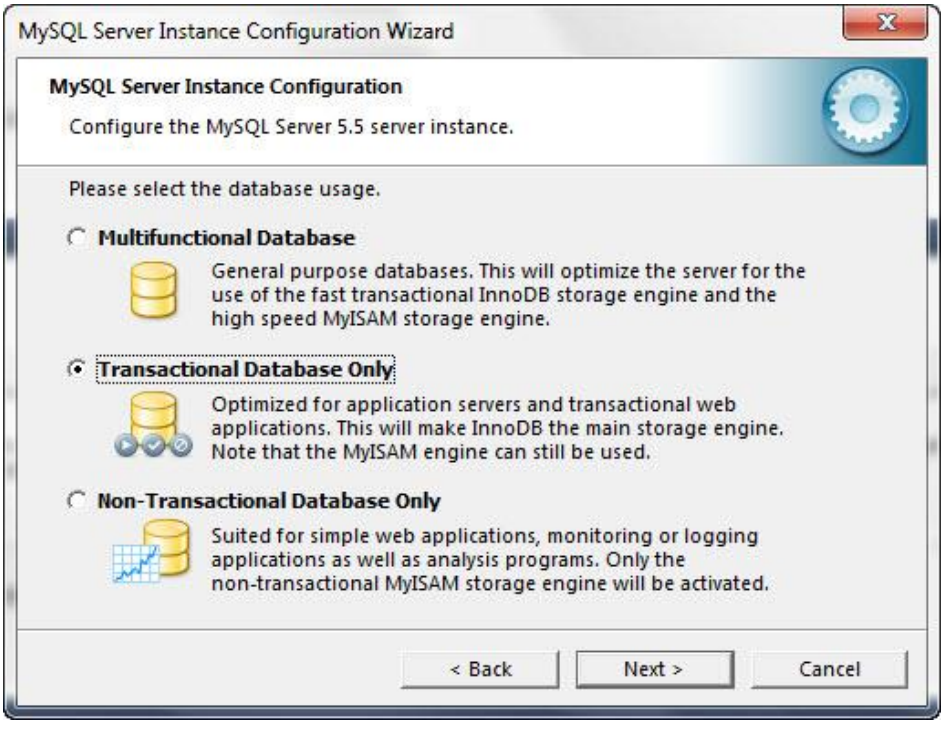

Ilustración 20. MySQL Server. Base de datos transaccional

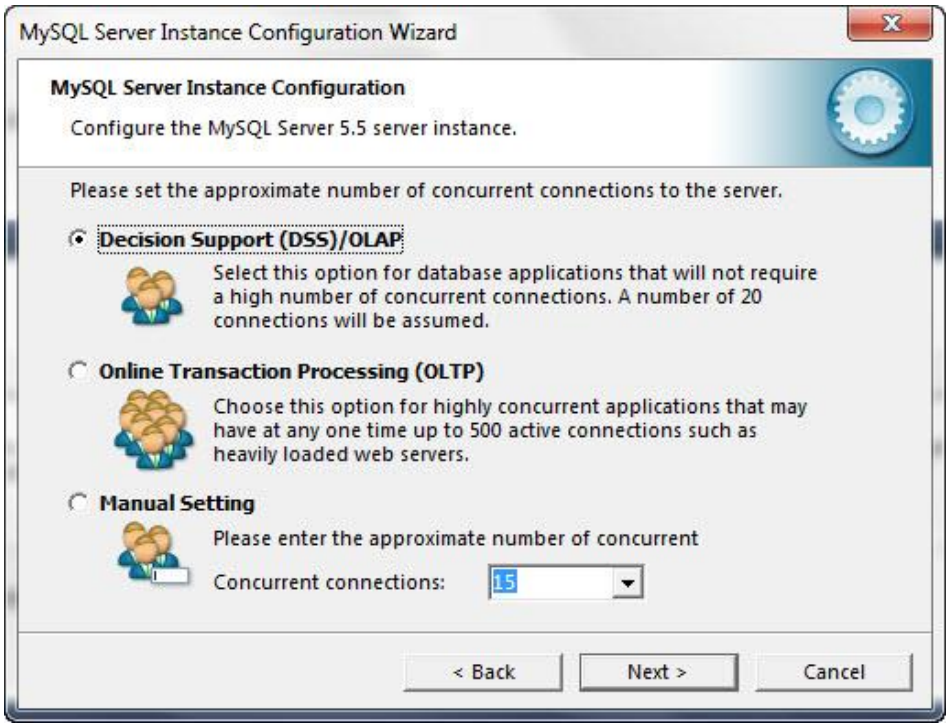

Ilustración 21. MySQL Server. Conexiones permitidas

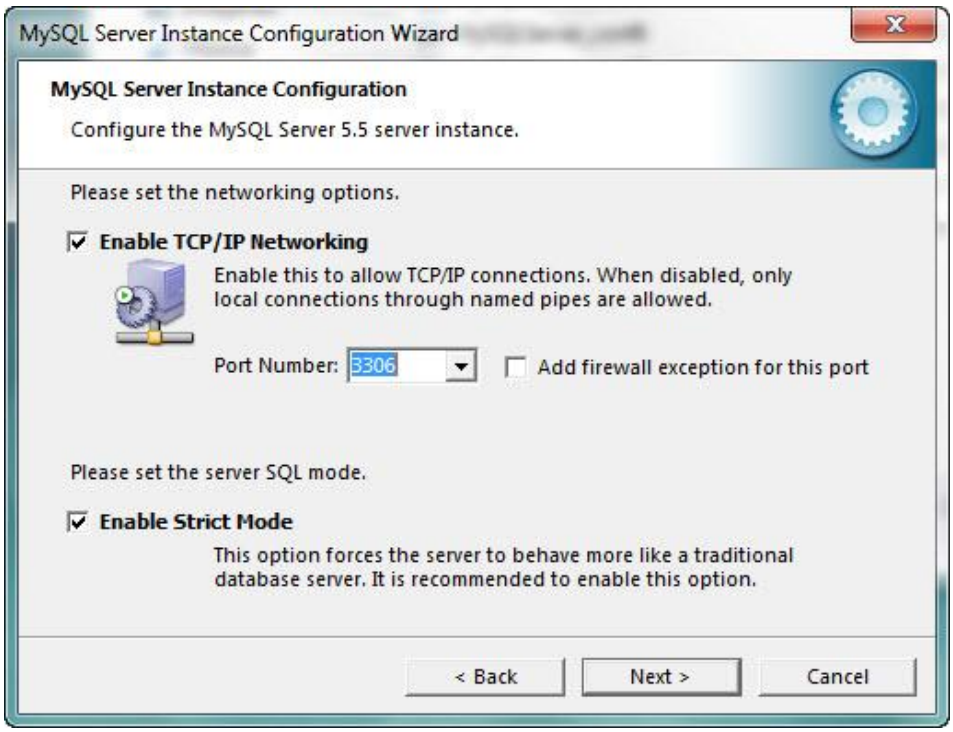

Ilustración 22. MySQL Server. Puerto de conexión

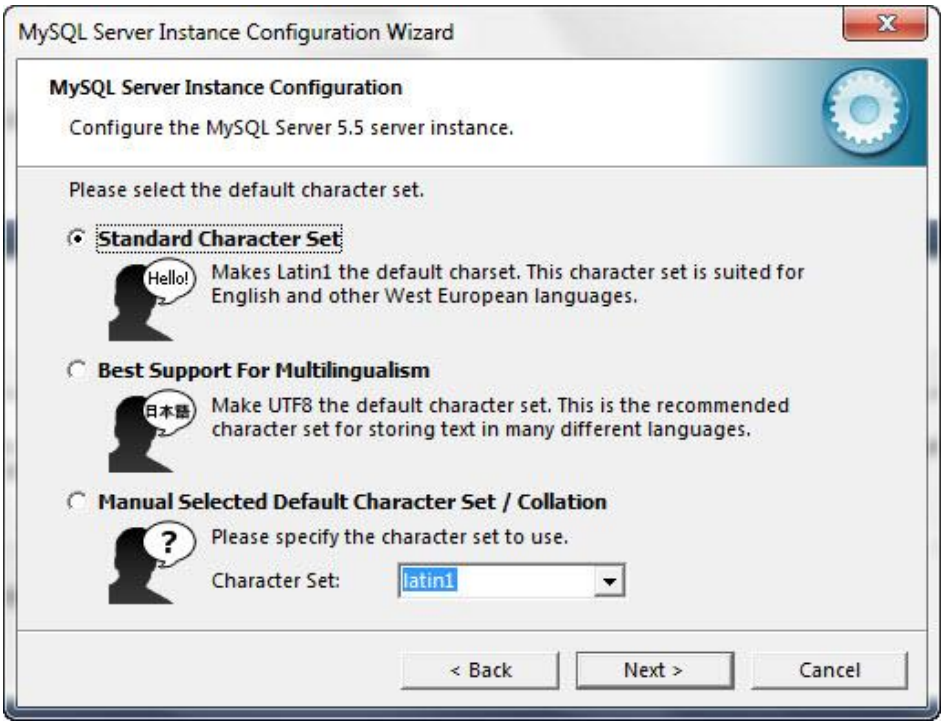

Ilustración 23. MySQL Server. Tipografía de caracteres

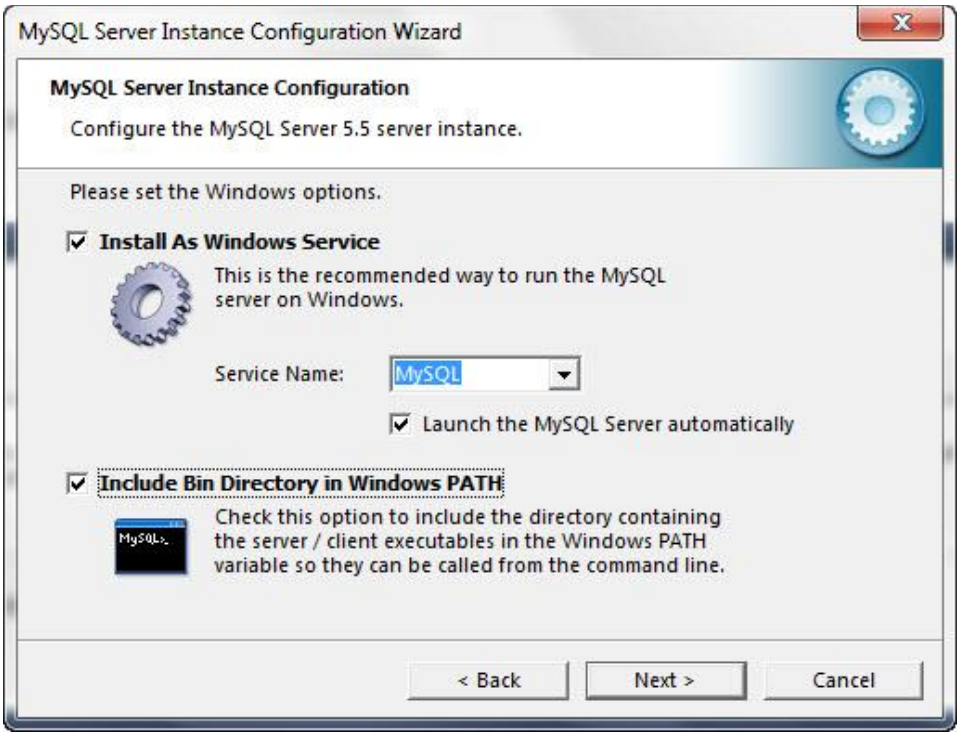

Ilustración 24. MySQL Server. Opciones de Windows

La contraseña que ponemos en la siguiente pantalla, será necesaria para poder acceder a la base de datos, tanto desde el código de la aplicación, como en MySQL Workbench.

Ilustración 25. MySQL Server. Contraseña de acceso

MySQL Workbench

Este programa, es una gran ayuda para gestionar las bases de datos. Además, dispone de herramientas que por ejemplo nos permite exportar los datos para poder trasladar la base de datos completa sin ningún problema, o generar el modelo de la base de datos que ya tenemos creada.

Crear base de datos

La primera vez que iniciamos el programa, nos aparecerá la instancia creada durante la instalación del servidor. Mediante esa conexión, podremos crear un nuevo esquema de base de datos, sobre la que ir creando tablas para almacenar los datos de nuestra aplicación web.

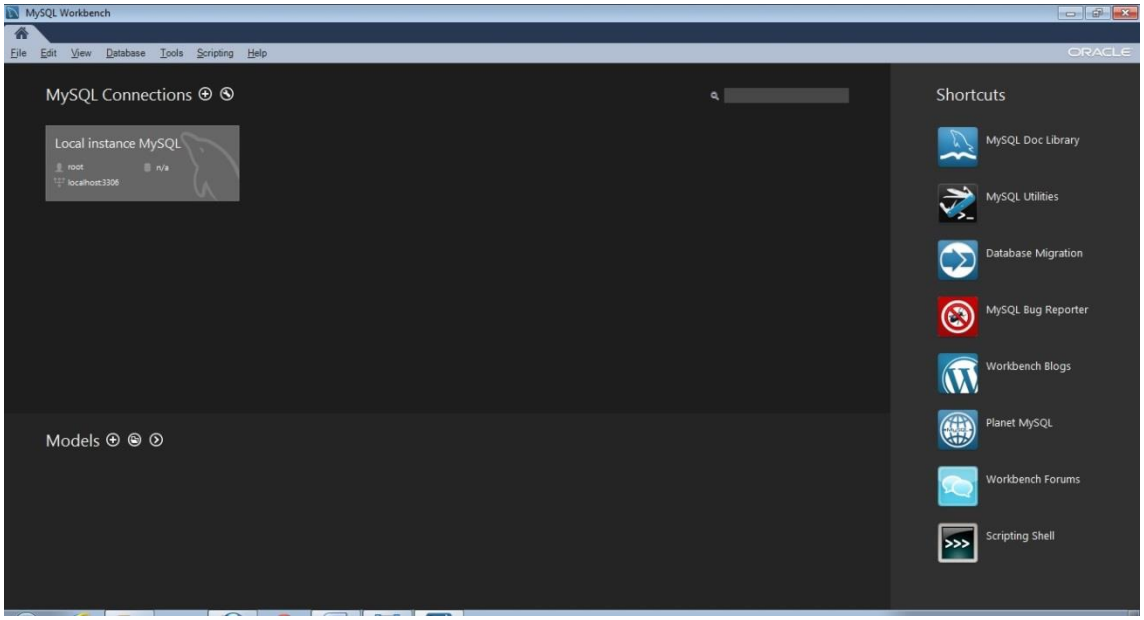

Ilustración 26. MySQL Workbench. Instancia de conexión

Para acceder, basta con hacer doble click, y nos solicitará la contraseña que pusimos en la configuración de la instancia.

Ilustración 27. MySQL Workbench. Seguridad de acceso

Tras introducirla, nos aparecerá la pantalla en la que se nos mostrará el centro de trabajo sobre la instancia abierta.

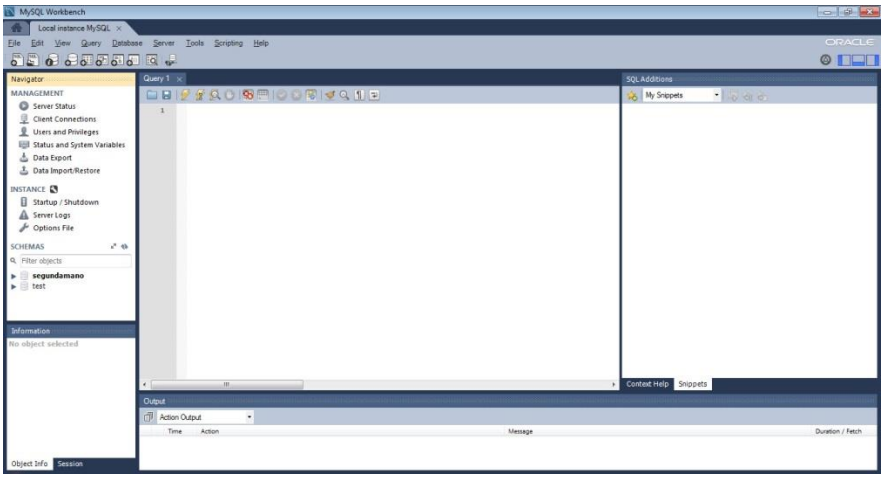

Ilustración 28. MySQL Workbench. Centro de trabajo

Para crear nuestra base de datos, nos dirigimos a la botonera superior, en la que solo necesitamos dos botones, señalados en rojo en la siguiente imagen. Pulsando sobre el símbolo de la izquierda, nos pedirá el nombre de la base de datos, y se creará el nuevo esquema. Una vez creado, con el segundo símbolo del cuadro rojo, iremos creando las tablas necesarias.

Ilustración 29. MySQL Workbench. Botonera de acciones

Exportar datos

Una vez que tenemos creado el esquema de nuestra base de datos, con las tablas y los datos, podemos extraer la base de datos para implantarla en otro equipo, de manera que no perdamos los datos que hay almacenados.

Para ello, seleccionamos “Data export” en el menú lateral izquierdo. Nos saldrá en la zona principal la pestaña de administración. Seleccionamos el esquema de la base de datos que queremos exportar, marcamos la opción “Export to self-contained file” y elegimos la ruta de destino.

Al continuar, se guardará un único fichero con extensión ‘.sql’ que contendrá toda la información de la base de datos, listo para importarlo en otro equipo.

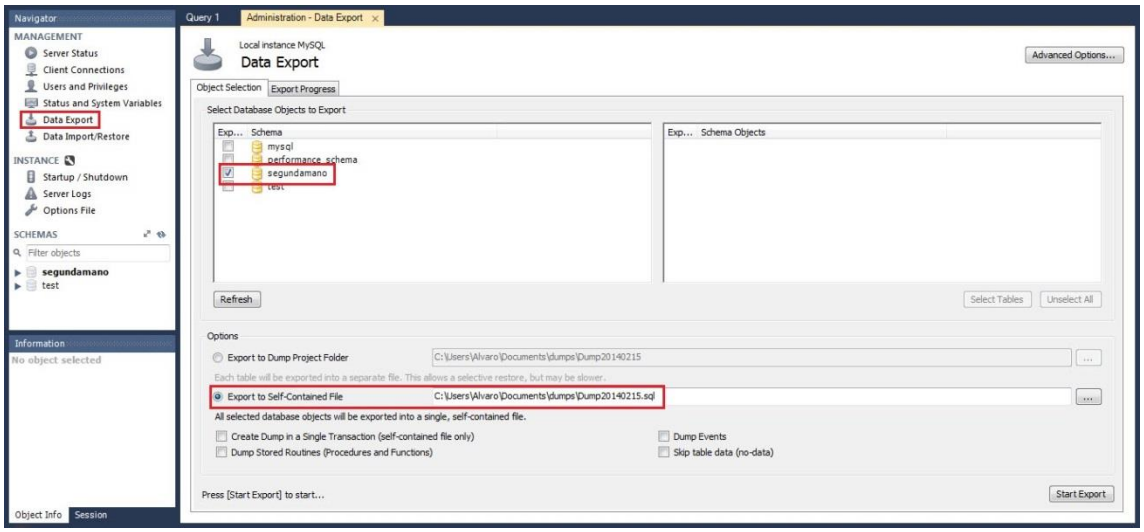

Ilustración 30. MySQL Workbench. Exportar base de datos